APPROACHES FOR EVALUATING INTERACTIONS IN A MULTI-PRODUCT INVENTORY SYSTEM

By
S. K. TANDON

IME
1996 TH
ME/1996/M
M TIS5a
TAN
APP

DEPARTMENT OF INDUSTRIAL & MANAGEMENT ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
JULY, 1976

APPROACHES FOR EVALUATING INTERACTIONS IN A MULTI-PRODUCT INVENTORY SYSTEM

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

By
S. K. TANDON

to the
DEPARTMENT OF INDUSTRIAL & MANAGEMENT ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
JULY, 1976

46897

AUG 1976

IME-1976-M-TAN-App

POST GRADUATE OFFICE
Submitted on 21.7.76

CERTIFICATE

This is to certify that the present work on 'Approaches for Evaluating Interactions in a Multi-Product Inventory System', by S.K. Tandon has been carried out under our supervision and has not been submitted elsewhere for the award of a degree.

P. Sharma

(P. Sharma)
Assistant Professor
Department of Mathematics
Indian Institute of Technology
Kanpur

J.L. Batra

(J.L. Batra)
Convener
Industrial and Management
Engineering Programme
Indian Institute of Technology
Kanpur

July, 1976

POST GRADUATE OFFICE
This thesis has been approved
for the award of the degree of
Master of Technology (M Tech.)
in accordance with the regulations of the Indian
Institute of Technology Kanpur
Dated: 30.7.76

ACKNOWLEDGEMENTS

I wish to express my deep sense of gratitude to Dr. (Mrs.) P. Sharma and Dr. J.L. Batra for their continued guidance and inspiration throughout the course of this work.

I stand indebted to Dr. A.K. Mittal for his help extended from time to time.

Thanks are due to so many of my friends who have helped me in moments of crisis, particularly Messrs. D.S. Londhe, G.K. Prasad and K.V. Iyer.

Credit for expert typing goes to Mr. J.K. Misra, his patience is gratefully acknowledged.

Sunil Kumar Tandon

TABLE OF CONTENTS

Chapter		Page
	LIST OF TABLES AND FLOW-CHARTS	vi
	SYNOPSIS	vii
1	INTRODUCTION	1
2	LITERATURE SURVEY	5
3	TWO PROBLEMS: THEIR FORMULATIONS AND SOLUTION PROCEDURES	10
	3.1 Problem 1	10
	3.1.1 Statement of the problem	10
	3.1.2 Assumptions	11
	3.1.3 Notations	12
	3.1.4 Problem Formulation	13
	3.1.5 Solution Procedure	18
	3.2 Problem 2	24
	3.2.1 Nomenclature	24
	3.2.2 Case of finite planning horizon	26
	3.2.3 Case of infinite planning horizon	34
4	METHODOLOGY TESTING AND SCOPE FOR FUTURE WORK	47
	4.1 Performance of Dynamic Programming Procedure	47
	4.2 Performance of Heuristic Algorithms	46
	4.3 Scope for Future Work	53
	REFERENCES	

Chapter | Page

APPENDIX

A Computer Listing for the Discrete Dynamic Programming

B Computer Listing for Heuristic 1

C Computer Listing for Heuristic 2

D Computer Listing for Heuristic 3

LIST OF TABLES AND FLOW-CHARTS

Table		Page
4.1	Average Computational Time for Problems Solved by Discrete Dynamic Programming Procedure	55
4.2	Data Used for a Three Item Inventory Problem Solved by Both Discrete and Continuous Dynamic Programming Procedures	55
4.3	Optimal Ordering Policies Obtained by Discrete Dynamic Programming Procedure for the Problem Given in Table 4.2.	56
4.4	Optimal Ordering Policies Obtained by Continuous Dynamic Programming Procedure for the Problem Given in Table 4.2.	56
4.5	Performance Comparison of Three Heuristics Based Cost Index Values	57
4.6	Performance Evaluation of the Three Heuristics Based on Average Computational Time.	58
4.7	Convergence Evaluation of Heuristic Three	59
4.8	Range of Percentage Cost Savings Obtained by Heuristic One as Compared to the Procedure in which All the Items are Ordered at Each Period.	62

FLOW-CHARTS

1	Flow Chart for Heuristic One	63
2	Flow Chart for Heuristic Two	64
3	Flow Chart for Heuristic Three	65

SYNOPSIS

The present work is based on two separately developed concepts of procurement and inventory operations. For these concepts, two models are proposed. The first model deals with a multi-item inventory system involving single source and limited warehouse capacity. A dynamic programming formulation with usual cost terms is structured. Objective is to minimize the total cost per period. The total cost is expressed as a function of maximum inventory level. Depending on the situation under consideration, this variable can take discrete or continuous values. Therefore, both discrete and continuous dynamic programming have been used to solve the problem.

The second model deals with joint replenishment of several items. The problem has been formulated mathematically with the objective of minimizing the total cost. Three heuristic approaches have been developed. Heuristic one considers a planning horizon of finite duration. Heuristics two and three solve the same problem but the planning horizon is taken as infinite. For an n-item problem heuristic two generates n feasible solutions, and the feasible solution giving the minimum objective function value isselected. Heuristic three is an iterative procedure and it generates only one solution.

The three heuristics have been compared for a set of 45 problems generated randomly. It is found that heuristic three which is computationally very efficient yields better solutions in most of the cases.

CHAPTER I

INTRODUCTION

The control and maintenance of inventories of physical goods is a problem common to all enterprises in any sector of a given economy. Inventories are maintained by manufacturing as well as non-manufacturing organizations. The economic significance of effective control of inventory is without question very great, for the costs associated with procurement and holding of inventory are large in almost all industrial, business and service organizations. It is no secret that inventories consume most of the investment capital in any enterprise. In fact inventories can represent more than 50 percent of a company's invested capital and as much as 80 percent of its cost of goods sold [10]. Because of the magnitude of costs involved and the widespread applicability of inventory problems, it is obvious that inventory theory is a fertile field for cost reduction efforts.

Although literature on inventory can be traced back to the turn of the century, useful work has ensued only after 1954. The later work attempts at structuring real life problems. The formulations, however, have been so complex and cumbersome that most of the research workers have concentrated on only single product inventory models.

The present work can be seen as consisting of two parts. The first part presents a model for developing the optimal procurement and inventory policies with respect to a cost function in a deterministic system composed of many items that may be procured from a specified source. The system is subject to a restriction on total warehouse space. The policy variables for the system under consideration include the procurement level and economic order quantity for each item. These are determined with respect to the relevant costs, system parameters, and a warehouse restriction such that the minimum total cost results. The minimization is performed through the application of dynamic programming technique. The solution proceeds stagewise (where each stage corresponds to an item) with the aid of recurrence relations and a functional equation method. Computer programs for both dynamic programming (discrete) and dynamic programming (continuous) have been developed.

The second part of the presentation aims at finding the optimal inventory policies (i.e. when to order and how much to order) with respect to a cost function when it is possible to replenish many items jointly. In case of joint replenishment the economic lot size for each item cannot be determined without considering the rest of the items within

the group. Each time a batch of product is procured, there is a common set-up cost for procurement in addition to the individual set-up cost for each of the items. Therefore, the economic order quantity for each item cannot be determined without considering the rest of theitems which are jointly replenished with the item in question.

The economic incentive behind thedesire for joint replenishment as a group lies in the fact that if each item is replenished individually, it must be responsible for the full set-up cost for each of its procurement run. In a manufacturing context, economies may accrue through joint replenishment as a consequence of savings in facility set-up and changeover time. In jointly ordering a number of items from a vendor economiesmay also accrue as a consequence of the preparation, processing and expediting of fewer orders and the receiving of fewer shipments.

To solve such problems one common approach is to determine independently the lot size for each item as if no inter-relationship among items exists, and then to coordinate the replenishment of the items by modifying the lot sizes and cycle times of the items. One approach used in practice is to decrease the cycle time (and, accordingly, the order quantity) of all the items with a low order frequency, making it equal to the highest frequency item. The shortcoming of

this approach, however, is that it is unreliable. The method does not always yield a solution with minimum total cost. In fact, it sometimes yields solutions which are more costly than those in which all cycle times are simply left at their original values.

In the following chapters, three different algorithms are presented for determining minimum cost solutions to multi item problems of this type. The first algorithm assumes a planning horizon of a finite duration and the last two algorithms assume a planning horizon of infinite duration. Solutions are sought in which all demands are met exactly and forwhich a reasonably good value of objective function is obtained.

CHAPTER II

LITERATURE SURVEY

Inventory control is a field which is rapidly growing in terms of available literature and an attempt to assess the work which has been done already in this field and placing it in its proper perspective appears to be an uphill task. Keeping in view the objective of the present work, this chapter concentrates on 'Multi-product Inventory Models'. An attempt has been made to give an account of the state of literature as it stands today.

Starr and Miller [23] appear to be the first to introduce the concept of constrained multi-product inventory systems. They assumed instantaneous replenishment and backorders were not permitted. With the use of Lagrangian multipliers, an optimal stocking policy was determined for a given set of physical and economic constraints. Problem of this type has also been investigated by Hansman [7], Buchan and Koenigsberg [3], and Hadley and Whitin [6].

Naddor [17] considered a two product system and found two optimal policies. The first policy pertained to the joint ordering of products using the classical lot-size formula. In the second policy independent ordering of items

was permitted. Naddor and Saltzmann [18] considered a multiproduct system where the objective function was a sum of carrying and ordering costs. However, the ordering cost was a function of items listed on a single order. The problem was solved using numerical approximations and it gave suboptimal results. Davis [4] modified the Naddor and Saltzman's approach. He formulated an objective function that consisted of ordering, interest, obsolescence and holding costs. The holding cost term included costs for warehouse space, labour and equipment. However, since the obsolescence, interest, and holding costs were all of the same form (a percent of the value of the item) they were consolidated into a single term that was similar to carrying costs considered in previous investigations. As before in the Naddor and Salzman model, the terms were independent and the total cost was simply the sum of costs for each item considered independently.

Plossl [20] proposed a LIMIT system to find optimal ordering policies, once the number of orders and reorder quantity are known. LIMIT is an acronym for Lot-size Inventory Management by Interpolation Technique. It takes into account the practical limitation on the total number of set-up hours which are available. Optimal order quantity is calculated by the Wilson's Lot Size formula and then the number of orders to

satisfy the demand are determined. By an iterative procedure the optimal mode ofordering is found for a given multi-product inventory problem. The method was designed to reduce the total ordering costs. The method, however, did not recognise any product interactions in the form of limited storage space or budgetory restriction.

Shu [22] has described a method for determining replenishment frequencies of two items jointly replenished. Essentially, the procedure developed a skipping rule for items. The skipped item is replenished every K-th time ($K \geq 1$) that the orders are placed. A set of graphs is provided to determine K and N (where N is the number of times the product will be procured). If the number of items exceeds two, Shu suggested that the items be ranked according to decreasing order of sales and that the smallest demand item be chosen as skipped item. The same procedure is to be repeated until the value of K for the item in question happens to be 1 or there are no more items to evaluate except the largest demand item. Karlo [13] has modified Shu's procedure. He suggested that items be ranked in the decreasing order of demand/set-up cost ratio and that the item with smallest ratio be selected as skipped item.

Andrews and Emmons [1] have presented a branch and bound algorithm for finding the optimal ordering policy for a multi-product system with deterministic demand and no backlogging. Ordering costs were assumed to include a separate set-up cost 'K_i' for each product ordered but with a fixed savings, K, each time all products were ordered simultaneously. The method, however, does not take care of the savings in cost when a few (not all) items are ordered at the same point.

Ignall [11] in his paper developed a method to determine minimum average cost ordering policies for continuously reviewed two product systems with joint set-up costs. Markov Renewal Programming was used to find the region in parameter space where a given policy was optimal.

Herron and Hawley [9] have considered a multiproduct inventory problem where the objective function includes a term for storage area cost which is a function of maximum expected inventory level Gerald F. Saxton [21] has considered extensively, product interactions due to storage area. Saxton's work consists of developing an analytic method for determining an optimum cost policy as a function of order quantity for a total cost multi-product inventory model where demands are known with certainity. The objective function is a sum of storage, replenishment, carrying,

obsolescence, area, bin, man-power and equipment costs. A computer simulation is included to varify input parameters as well as the resultant system cost as provided by the analytic model.

Lastly, Arvind R. Lele [15] has proposed a model for a multiproduct inventory problem with budgetary and storage space limitations. Lele structures the situation as a non-linear programming problem. The problem is solved by Davidon-Fletcher-Powell Variable Metric Method with Golden Search technique applied for uni-directional minimisation.

The review of pertinent literature in the field of inventory control has uncovered several aspects that merit further attention. One of these areas concerns the interactions which are present in a multi product system. It was, therefore, felt that solutions making partial incision into this area would be an addition to this field.

The first part of the present study deals with a multi product inventory problem where there is an upper limit imposed on total available storage space and items are competing for the floor space. The second part of the study investigates a problem where product interactions are present due to joint replenishment of two or more items.

CHAPTER III

TWO PROBLEMS: THEIR FORMULATIONS AND SOLUTION PROCEDURES

In this chapter we consider two multi-item inventory problems, and give their solution procedures. The first part of the chapter presents a dynamic programming algorithm to find out optimal ordering policies for a multi item inventory system subject to warehouse capacity constraint. In the second, joint replenishment of several items at a time is considered and three heuristic procedures are developed to solve this problem.

3.1 Problem 1:

A statement of the problem is given below:

3.1.1 Statement of the problem:

A brief description of the decision situation under consideration is given below:

Several items are maintained in inventory to meet the demands. In satisfying these demands, the systems manager finds it necessary to replenish his stock for each item periodically. A limited warehouse space is available for stocking the items. The manager must decide

when to procure each item and how much of each item to procure so that at no time the inventory level exceeds the maximum permissible by the available warehouse space. The decision environmentis composed of item dependent parameters. The objective is to determine the procurement level and the procurement quantity for each item so that the sum of all the costs associated with the inventory system under consideration is minimized.

3.1.2 Assumptions:

The following simplifying assumptions have been made in formulating the above problem:

1. The demand rate, replenishment rate and lead time for each item are known and constant.

2. Backorders are permitted.

3. Item cost, procurement cost, holding cost and back-order cost for each item are known and constant.

4. Only a limited warehouse space is available.

5. The maximum inventory level in the warehouse is sum of the maximum inventory levels for individual items.

3.1.3 Notations:

The following notations are used to define the problem:

TSC = total system cost per period

TC_i = total cost of item 'i' per period

TC_i^* = minimum total costof item i per period

HC_i = holding cost of item i per period

BC_i = backorder cost of item i per period

PC_i = procurement cost of item i per period

C_i = unit cost of item i.

d_i = demand rate for item i

r_i = replenishment rate for item i

IC_i = total unit cost of d_i items

h_i = cost of holding one unit of item i in stock per period

b_i = cost of having one unit of item i backlogged per period

p_i = cost of procurement of item i per procurement

m_i = number of periods per inventory cycle of item i

t_{Li} = lead time for item i

Q_i = order quantity for item i

Q_i^* = optimum order quantity for item i

L_i = procurement level for item i

L_i^* = optimum procurement level for item i

I_i = maximum inventory level in an inventory cycle of item i

B_i = maximum backorder level in an inventory cycle of item i

W = total available warehouse space

w_i = warehouse space consumed by one unit of item i

3.1.4 Problem formulation:

Since both demand and lead time for all the items are assumed deterministic, the resulting inventory process will be as indicated in Figure 1. The exhibited geometry of such a process will depend on procurement level, L_i, the procurement quantity, Q_i, the demand rate, d_i, the procurement lead time, t_{Li} and the replenishment rate, r_i. Since these parameters are item dependent, it is evident that the geometry will be item dependent.

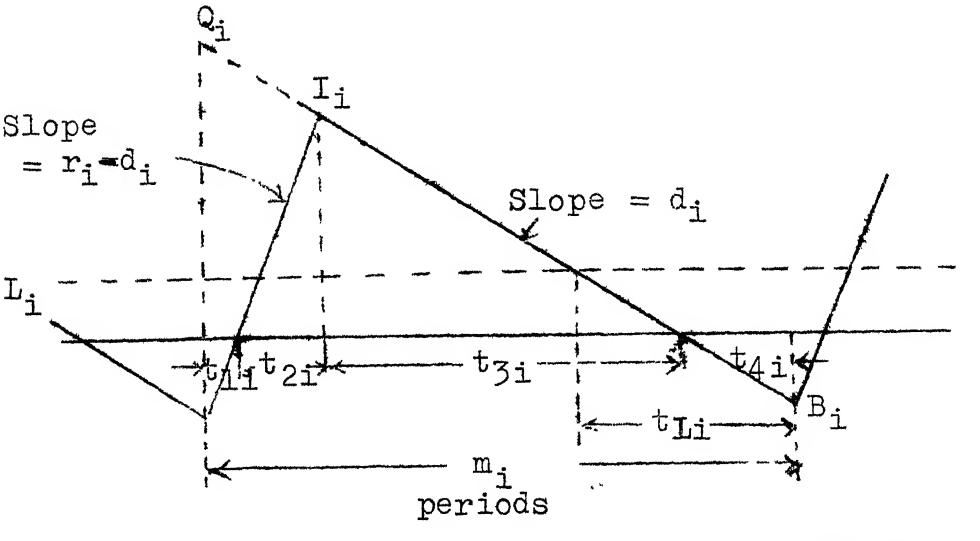


Fig. 1: Deterministic Inventory Process.

Mathematical Relationships:

From Figure 1, it is evident that the number of periods per inventory cycle for item i are:

$$m_i = \frac{Q_i}{d_i} \tag{1}$$

Also, the following relationships are evident:

$$(t_{1i} + t_{2i})\ (r_i - d_i) = (t_{3i} + t_{4i})\ d_i \tag{2}$$

$$t_{1i} + t_{2i} = \frac{Q_i - (t_{1i} + t_{2i})\ (r_i - d_i)}{d_i}$$

or

$$t_{1i} + t_{2i} = \frac{Q_i}{r_i} \tag{3}$$

and

$$t_{3i} + t_{4i} = \frac{I_i + d_i \cdot t_{Li} - L_i}{d_i} \tag{4}$$

From equations (2), (3) and (4), we get,

$$I_i = Q_i\ (1 - \frac{d_i}{r_i}) + L_i - d_i \cdot t_{Li} \tag{5}$$

The total time per cycle during which inventory is held is $(t_{2i} + t_{3i})$. This is given by,

$$t_{2i} + t_{3i} = \frac{I_i}{r_i - d_i} + \frac{I_i}{d_i} \tag{6}$$

From equations (5) and (6), we get,

$$t_{2i} + t_{3i} = [Q_i (1 - \frac{d_i}{r_i}) + L_i - d_i . t_{Li}](\frac{1}{r_i - d_i} + \frac{1}{d_i}) \quad (7)$$

We observe the following relationship:

$$\frac{1}{r_i - d_i} + \frac{1}{d_i} = \frac{1}{d_i (1 - \frac{d_i}{r_i})} \quad (8)$$

From equations (7) and (8), we get,

$$t_{2i} + t_{3i} = [Q_i (1 - \frac{d_i}{r_i}) + L_i - d_i . t_{Li}] \frac{1}{d_i (1 - \frac{d_i}{r_i})} \quad (9)$$

The total time for which a backorder exists is $(t_{1i} + t_{4i})$. This is given by:

$$t_{1i} + t_{4i} = \frac{B_i}{r_i - d_i} + \frac{B_i}{d_i} \quad (10)$$

But, since $B_i = d_i . t_{Li} - L_i$, we get:

$$t_{1i} + t_{4i} = (d_i . t_{Li} - L_i) (\frac{1}{r_i - d_i} + \frac{1}{d_i}) \quad (11)$$

From equations (8) and (11), we get the following relationship:

$$t_{1i} + t_{4i} = (d_i . t_{Li} - L_i) \frac{1}{d_i (1 - \frac{d_i}{r_i})} \quad (12)$$

Total System Cost:

The total cost per period for each item will be a summation of the item cost per period, the procurement cost per period, the holding cost per period and the back-order cost per period, that is:

$$TC_i = IC_i + PC_i + HC_i + BC_i \qquad (13)$$

The item cost per period will be the product of the item cost per unit and the demand rate in units per period,

$$IC_i = C_i\, d_i \qquad (14)$$

The procurement cost per period will be the procurement cost per procurement divided by the number of periods per inventory cycle,

$$PC_i = \frac{p_i}{m_i}$$

$$= \frac{p_i \cdot d_i}{Q_i} \qquad (15)$$

Holding cost per period will be the product of the holding cost per unit per period and the average number of units on hand during the period,

$$HC_i = \frac{1}{2} \frac{h_i\,(t_{2i} + t_{3i})\, I_i}{m_i}$$

Substituting for $(t_{2i} + t_{3i})$, I_i and m_i in the above equation, we get:

$$HC_i = \frac{h_i}{2Q_i\left(1 - \frac{d_i}{r_i}\right)}\left[Q_i\left(1 - \frac{d_i}{r_i}\right) + L_i - d_i \cdot t_{Li}\right]^2 \qquad (16)$$

Backorder cost per period will be the product of the backorder cost per unit per period and the average number of units backordered during the period,

$$BC_i = \frac{1}{2}\,\frac{b_i\,(t_{1i} + t_{4i}).B_i}{m_i}$$

or,

$$BC_i = \frac{b_i\,(d_i \cdot t_{Li} - L_i)^2}{2Q_i\left(1 - \frac{d_i}{r_i}\right)} \qquad (17)$$

The total cost per period will be a summation of the four cost components given by equations (14), (15), (16) and (17). Therefore,

$$TC_i = C_i \cdot d_i + \frac{p_i \cdot d_i}{Q_i} + \frac{h_i}{2Q_i\left(1 - \frac{d_i}{r_i}\right)}\left[Q_i\left(1 - \frac{d_i}{r_i}\right) + L_i - d_i \cdot t_{Li}\right]^2 + \frac{b_i\,(L_i - d_i \cdot t_{Li})^2}{2Q_i\left(1 - \frac{d_i}{r_i}\right)} \qquad (18)$$

The total system cost will be given by,

$$TSC = \sum_{i=1}^{n} TC_i \qquad (19)$$

Warehouse Capacity Constraint:

The i-th item in the inventory system consumes certain amount of warehouse space, w_i. There exists a certain amount of total warehouse capacity, wW. The maximum accumulation of inventory for the i-th item, I_i, will consume $I_i w_i$ units of warehouse space. Therefore, the restriction $\sum_{i=1}^{n} I_i w_i \leq W$ must not be violated. In the following section, a dynamic programming algorithm is presented that will find the optimal procurement and inventory policies in the face of this restriction.

3.1.5 Solution procedure:

The first step in finding the optimal procurement and inventory policies for the inventory system, is to determine whether the constraint is active or not. Below, we give the procedure for finding the optimal policies when the constraint is active as well as when it is not active.

1. Optimal Policies when Constraint is Inactive:

The minimum cost procurement level and procurement quantity for each item may be found by setting the partial derivatives equal to zero and solving the resulting equations. From equation (18) we obtain:

$$TC_i = C_i.d_i + \frac{p_i.d_i}{Q_i} + \frac{h_i.Q_i\ (1 - \frac{d_i}{r_i})}{2} - h_i\ (d_i.t_{Li} - L_i)$$

$$+ \frac{h_i\ (d_i.t_{Li} - L_i)^2}{2Q_i\ (1 - \frac{d_i}{r_i})} + \frac{b_i\ (d_i.t_{Li} - L_i)^2}{2Q_i\ (1 - \frac{d_i}{r_i})} \qquad (20)$$

Taking the partial derivative with respect to Q_i, then with respect to $(d_i.t_{Li} - L_i)$ and setting both equal to zero give :

$$\frac{\partial TC_i}{\partial Q_i} = -\frac{p_i.d_i}{Q_i^2} + \frac{h_i\ (1 - \frac{d_i}{r_i})}{2} - \frac{h_i(d_i.t_{Li} - L_i)^2}{2Q_i^2\ (1 - \frac{d_i}{r_i})}$$

$$- \frac{b_i\ (d_i.t_{Li} - L_i)^2}{2Q_i^2\ (1 - \frac{d_i}{r_i})} = 0 \qquad (21)$$

and,

$$\frac{\partial TC_i}{\partial (d_i.t_{Li} - L_i)} = -h_i + \frac{h_i\ (d_i.t_{Li} - L_i)}{Q_i\ (1 - \frac{d_i}{r_i})}$$

$$+ \frac{b_i\ (d_i.t_{Li} - L_i)}{Q_i\ (1 - \frac{d_i}{r_i})} = 0 \qquad (22)$$

Equation (22) can be written as:

$$\frac{d_i t_{Li} - L_i}{Q_i} = \frac{h_i\ (1 - \frac{d_i}{r_i})}{h_i + b_i} \qquad (23)$$

Substituting equation (23) into (21), we get,

$$Q_i^* = \left[\frac{2p_i \cdot d_i}{\left(1 - \frac{d_i}{r_i}\right)} \left(\frac{1}{h_i} + \frac{1}{b_i} \right) \right]^{\frac{1}{2}} \qquad (24)$$

Substituting equation (24) into equation (23) gives:

$$L_i^* = d_i \cdot t_{Li} - \left[\frac{2p_i \cdot d_i \left(1 - \frac{d_i}{r_i}\right)}{b_i \left(1 + \frac{b_i}{h_i}\right)} \right]^{\frac{1}{2}} \qquad (25)$$

Equation (24) and Equation (25) may now be substituted back into equation (20) to give an expression for the minimum cost. After several simplying steps, we obtain:

$$TC_i^* = C_i \cdot d_i + \left[\frac{2 \left(1 - \frac{d_i}{r_i}\right) p_i \, d_i \, h_i \, b_i}{(h_i + b_i)} \right]^{\frac{1}{2}} \qquad (26)$$

2. Optimal Policies when Constraint is Active:

Since I_i consumes warehouse space it is necessary to express TC_i for each valueof I_i over a reasonable range. These values make up cost functions for thedynamic programming algorithm.

Equation (5) may be solved for $(d_i \cdot t_{Li} - L_i)$ giving:

$$(d_i \cdot t_{Li} - L_i) = Q_i \left(1 - \frac{d_i}{r_i}\right) - I_i \qquad (27)$$

Substituting equation (5) and equation (27) into equation (18) gives:

$$TC_i = C_i \cdot d_i + \frac{p_i \cdot d_i}{Q_i} + \frac{h_i \cdot I_i^2}{2Q_i \left(1 - \frac{d_i}{r_i}\right)} + \frac{b_i \left[Q_i \left(1 - \frac{d_i}{r_i}\right) - I_i\right]^2}{2Q_i \left(1 - \frac{d_i}{r_i}\right)} \qquad (28A)$$

or

$$TC_i = C_i \cdot d_i + \frac{p_i \cdot d_i}{Q_i} + \frac{h_i \cdot I_i^2}{2Q_i \left(1 - \frac{d_i}{r_i}\right)} + \frac{b_i \left(1 - \frac{d_i}{r_i}\right)}{2} Q_i - b_i \cdot I_i + \frac{b_i \cdot I_i^2}{2Q_i \left(1 - \frac{d_i}{r_i}\right)} \qquad (28B)$$

Taking the partial derivative of TC_i with respect to Q_i and setting equal to zero we get:

$$\frac{\partial TC_i}{\partial Q_i} = -\frac{p_i \cdot d_i}{Q_i^2} - \frac{h_i I_i^2}{2Q_i^2 \left(1 - \frac{d_i}{r_i}\right)} + \frac{b_i \left(1 - \frac{d_i}{r_i}\right)}{2} - \frac{b_i \cdot I_i^2}{2Q_i^2 \left(1 - \frac{d_i}{r_i}\right)} = 0$$

This gives:

$$Q_i^* = \frac{1}{(1 - \frac{d_i}{r_i})} \left[\frac{2\, p_i . d_i\, (1 - \frac{d_i}{r_i}) + I_i^2\, (b_i + h_i)}{b_i} \right]^{\frac{1}{2}} \tag{29}$$

From equations (5) and (24), we get,

$$L_i^* = I_i + d_i . t_{Li} - \left[\frac{2\, p_i . d_i\, (1 - \frac{d_i}{r_i}) + I_i^2\, (h_i + b_i)}{b_i} \right] \tag{30}$$

Equations (29) and (30) give the minimum cost Q_i and the minimum cost L_i as a function of I_i and other parameters. The minimum cost may be expressed as a function of I_i and other parameters by substituting (29) and (30) into (28A) and modifying the last term. This gives:

$$TC_i^* = C_i . d_i + \frac{p_i . d_i}{Q_i^*} + \frac{h_i\, I_i^2}{2Q_i^* (1 - \frac{d_i}{r_i})} + \frac{b_i\, (d_i . t_{Li} - L_i^*)^2}{2Q_i^* (1 - \frac{d_i}{r_i})} \tag{31}$$

Equations (29), (30) and (31) may now be used to develop cost functions needed by the dynamic programming algorithm. In the dynamic programming algorithm presented

here, each stage corresponds to an item. Let $g_i\ (I_i w_i)$ represent the total cost when inventory level is I_i and let $f_i(W)$ represent the minimum cost value at the i-th stage (i.e. i-th item) when warehouse space available is 'W' (W is the state variable).

The solution proceeds stage-wise with the aid of recurrence relations technique. First, the cost expected from the first stage (item) if all warehouse space is allocated to it is determined from $f_1(W) = g_1(I_1 w_1)$. The computations for $f_1(W)$ are completed and results are entered in the first stage of the solution table. In general, for the j-th stage, following recurrence relation is used:

$$f_j(W) = \min_{0 \leq I_j w_j \leq W} [g_j\ (I_j w_j) + f_{j-1}(W - I_j w_j)] \qquad (32)$$

The results for each stage are entered in the solution table. The solution table is then used to find the optimal policies forthis constrained inventory system. The minimum total system cost is searched in the last stage of the solution table and corresponding inventory level is noted. Then we trace back through all the **stages** and inventory level at each stage (item) are noted. The optimal values of inventory levels so found i.e., I_i's, along with the equations (29) and (30) can be used to find out optimal order policies for this inventory system.

3.2 Problem 2:

The problem formulation and solution procedures for a multi-product inventory system with joint replenishment of products are presented in this section. Three heuristic procedures have been developed to handle both the cases when planning horizon is finite as well as when it is infinite.

3.2.1 Nomenclature:

The notations used in this section are given below:

n = total number of items

N = total number of set-ups in the planning horizon (for heuristic 1)

= total number of set-ups per year (for heuristics 2 and 3)

N^* = economic number of set-ups per year

N_i = number of set-ups for i-th item in the planning horizon (for heuristic 1)

= number of set-ups per year for i-th item (for heuristics 2 and 3)

N_i^* = economic number of set-ups for item i in the planning horizon (for heuristic 1)

= economic number of set-ups per year for item i (for heuristics 2 and 3)

k_i = relative ordering frequency for item i ($k_i = \frac{N}{N_i}$)

k_i^* = economic value of k_i

k_{ji} = relative ordering frequency for item j when item i is ordered at each set-up

k^c_{ji} = corrected value of k_{ji}

k_{ir} = relative ordering frequency of item i at the r-th iteration

k^c_{ir} = corrected valueof k_{ir}

T = planning horizon

P = number of periods in the planning horizon

m = number of time units per year

t_i = time duration in an inventory cycle for which a unit of item i is backlogged

Q_i = order quantity for item 'i'

Q^*_i = economic order quantity for item i

D_i = annual demand for item i

d_i = demand rate for item i

S = fixed cost of taking a set-up

s_i = variable cost of taking a set-up for item i

h_i = cost of holding one unit of item i for unit time

b_i = cost of having backlogged one unit of item i for unit time

w_i = warehouse space axoccupied by one unit of item i

W = total available warehouse space

R = cost of occupying one unit of warehouse space during the planning horizon (for heuristic 1)

= annual cost of occupying one unit of warehouse space (for heuristics 2 and 3)

VC_i = variable cost for item i in the planning horizon (for heuristic 1)

= annual variable cost for item i (for heuristics 2 and 3)

TC = total cost of the inventory system in the planning horizon (for heuristic 1)

= annual total cost of the inventory system (for heuristics 2 and 3)

TC_i = annual total cost of the inventory system when item i is ordered at each set-up

3.2.2 Case of Finite Planning Horizon:

1. Problem Statement:

There are n items stocked in a warehouse. The various costs, namely,

(i) Fixed and variable set-up costs,

(ii) Holding cost, and

(iii) Backorder cost

are known for each item. Decisions as to how many orders are to be placed, when to place them and how much to order each time are to be made. These decisions should minimize the total cost incurred over the finite planning horizon, T.

2. Assumptions:

The model developed, is based on a set of assumptions. The purpose of these assumptions is to simplify the analysis

of the problem. These assumptions are described below:

1. There is a family of n products which can be ordered together.
2. An item is replenished at equal time intervals.
3. Backorders are permitted.
4. Replenishment is instantaneous for each item.
5. There is only a limited storage space available for stock holding. However, additional space can be rented at a constant rate. If there is some extra space it can be rented out at the same rate.
6. The planning horizon is finite and known.
7. Demand rate for each item is given and is constant over time.
8. The demand for each item within the horizon period is a quantity greater than zero.
9. There will be at least one set-up for each item within the horizon period.
10. The planning horizon consists of a finite number of periods and a set-up can be taken up only at the commencement of a period.
11. An order will be placed for each item at the beginning of the first period.

13. The number of inventory cycles for each item, within the planning horizon, is an integer.

3. Problem Formulation:

The inventory process resulting from the problem as stated above is shown in Figure 2.

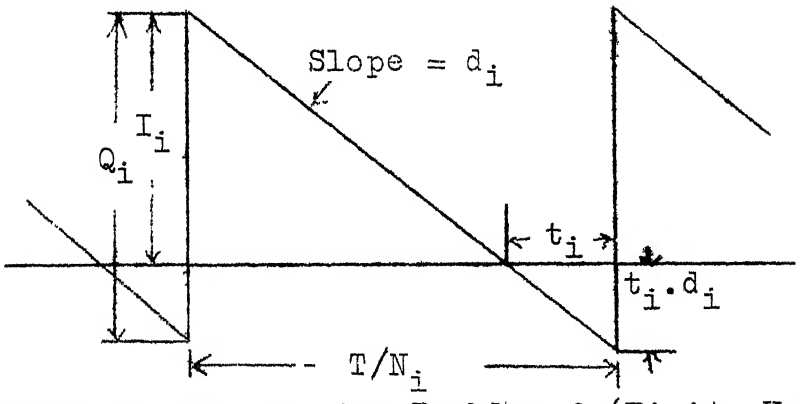


Fig.2: Inventory Process for Problem 2 (Finite Horizon)

Since the i-th item is replenished in N_i equally spaced set-ups in the horizon period, the variable cost for this item consists of:

(i) variable set-up cost of N_i set-ups;

(ii) total stock holding cost in the planning horizon;

(iii) total backorder cost in the planning horizon; and

(iv) cost due to warehouse space occupied.

Thus, we can write,

$$VC_i = N_i.s_i + \tfrac{1}{2}\,(Q_i - t_i.d_i)\left(\frac{T}{N_i} - t_i\right) N_i.h_i$$

$$+ \tfrac{1}{2}\, t_i^2.d_i.N_i.b_i + (Q_i - t_i.d_i)\, w_i.R \qquad (33)$$

The total cost of the inventory system can be written as,

$$TC = NS + \sum_{i=1}^{n} N_i s_i + \tfrac{1}{2} \sum_{i=1}^{n} (Q_i - t_i.d_i)\, x$$

$$\left(\frac{T}{N_i} - t_i\right) N_i.h_i + \tfrac{1}{2} \sum_{i=1}^{n} t_i^2.d_i.N_i.b_i$$

$$+ R\left(\sum_{i=1}^{n} (Q_i - t_i.d_i)\, w_i - W\right) \qquad (34)$$

where the order quantity, Q_i, is given by

$$Q_i = \frac{Td_i}{N_i} \qquad (35)$$

To find the optimum value of t_i which corresponds to minimum variable cost, substitute the value of Q_i in equation (33) and equate the partial derivative of VC_i with respect to t_i to zero. This gives,

$$t_i = \frac{w_i.R + T.h_i}{N_i(h_i + b_i)} \qquad (36)$$

Substituting this value of t_i in equations (33) and (34), and replacing Q_i by Td_i/N_i, we get,

$$VC_i = N_i.s_i + \frac{T^2 d_i}{2N_i} \cdot \frac{h_i, b_i}{h_i + b_i} + R \frac{w_i.d_i\ (2T.b_i - w_i.R)}{2N_i\ (h_i + b_i)} \qquad (37)$$

and

$$TC = NS + \sum_{i=1}^{n} [N_i.s_i + \frac{T^2 d_i}{2N_i} \cdot \frac{h_i.b_i}{h_i+b_i} + R \frac{w_i.d_i(2T.b_i - w_i.R)}{2N_i\ (h_i + b_i)}] - R.W \qquad (38)$$

4. Solution Procedure:

Heuristic Algorithm 1:

The solution procedure consists of two parts. The first part consists of finding the values of N_i for each item. In the second part, these values of N_i are plotted and all the points at which one or more items are ordered are counted. This gives the total number of set-ups, in the planning horizon.

The economic ordering frequency, N_i, for the i-th item corresponds to the minimum variable cost for the i-th item. The values of N_i for i-th item can, therefore, be obtained by taking the first derivative of equation (37) with respect to N_i and equating it to zero. This gives,

$$N_i = \left[\frac{T^2 . d_i . h_i . b_i + R . w_i . d_i (2T . b_i - w_i . R)}{2 s_i (h_i + b_i)} \right]^{\frac{1}{2}} \quad (39)$$

This value of N_i is then corrected to ensure an integer number of inventory cycles in the planning horizon.

A step by step algorithm of the proposed heuristic is given below:

Part 1: This part consists of obtaining the values of N_i.

Step 1: From the given value of P find out the possible values which N_i can take. Since the number of set-ups, N_i, for the i-th item should be an integer, the possible values which N_i can take will be the factors of P. Let these be denoted by $a_1, a_2 \ldots a_p, \ldots a_m$. Thus, for each item i the number of set-ups for that item will be given by one of the a_p's. To illustrate this, let P = 12, then the values of a_p will be $a_1 = 1$, $a_2 = 2$, $a_3 = 3$, $a_4 = 4$, $a_5 = 6$ and $a_6 = 12$. So N_i can take any of the values 1, 2, 3, 4, 6 and 12.

Step 2: Find out the value of N_i for i-th item from equation (39).

Step 3: If this value of N_i is less than one then put $N_i^* = 1$ and go to Step 9, otherwise continue.

Step 4: If the value of N_i calculated in Step 2 is greater than P, then put $N_i^* = P$ (This is done because there

cannot be more than P set-ups in the planning horizon) and go to Step 9 otherwise continue.

Step 5: If the value of N_i is equal to one of the factors of P, say a_p, then put $N_i = a_p$ and go to Step 9, otherwise continue.

Step 6: If the value of N_i lies between a_j and a_{j+1} then put $N_i' = a_j$ and compute the value of variable cost VC_{i1}, for i-th item by the following formula:

$$VC_{i1} = N_i' . s_i + \frac{T^2 . d_i}{2N_i'} \frac{h_i . b_i}{h_i + b_i} + R \frac{w_i . d_i \ (2T . b_i - w_i . R)}{2N_i' \ (h_i + b_i)}$$

Step 7: Take the next higher value of a i.e. a_{j+1} and let $N_i'' = a_{j+1}$. Again compute the variable cost with this value of N_i''. Let it be VC_{i2},

$$VC_{i2} = N_i'' . s_i + \frac{T^2 . d_i}{2N_i''} \frac{h_i . b_i}{h_i + b_i} + R \frac{w_i . d_i \ (2T . b_i - w_i . R)}{2N_i'' \ (h_i + b_i)}$$

Step 8: If the value of VC_{i1} is less than or equal to VC_{i2} then put $N_i = N_i'$ and go to Step 9,

otherwise put $N_i = N_i''$.

Step 9: Repeat Steps 2 through 8 for all items. If the values of N_i for all the items have been calculated then go to Step 10.

Part 2: This part of the algorithm finds out the total number of set-ups, N, thetotal cost of the system and the economic order quantity for each item.

Step 10: Using the above calculated values of N_i, find out R_i for the i-th item where $R_i = P/N_i$.

Step 11: An order for every item is placed at the first period and successive orders are placed after every R_i periods. Find out the periods at which an order is placed for the i-th item. Let these be denoted by $p_{i1}, p_{i2}, \ldots, p_{ik}, \ldots p_{il}$ (where the maximum value of p_{ik} can be P and minimum value can be 1).

Step 12: Repeat Step 10 and Step 11 for all the items and find out the values of p_{ik}'s for all the items.

Step 13: Count those periods at which one or more items are ordered. The total number of such periods give the total number of set-ups, N.

Step 14: Obtain the economic order quantity for i-th item from the following formula:

$$Q_i^* = \frac{Td_i}{N_i^*}$$

Step 15: Obtain the total costof the system by substituting values of N and N_i^*, in the equation (38).

3.2.3 Case of Infinite Planning Horizon:

In this section two procedures are described which deal with a joint replenishment problem of a multi-item inventory system when the planning horizon is infinite. The statement of the problem is given below:

1. Problem Statement:

In the previous model we treated the planning horizon as finite. The same problem is dealt here with an infinite planning horizon. The decisions are to be made such that the total cost per year is minimum.

2. Assumptions:

The first five assumptions made in the previous model hold true for this case also. Following additional assumptions have been made:

1. The set-ups are taken at equal time intervals.
2. Planning horizon is either infinite or it is so long compared to the cycle time that it can betaken as infinite.

3. Atleast one item is ordered at every set-up.
4. Total number of set-ups per year need not be an integer.
5. All items are ordered at the first set-up.
6. Annual demand for each item is a quantity greater than zero.

3. Problem Formulation:

The inventory process resulting from the problem statement given above is shown in Figure 3.

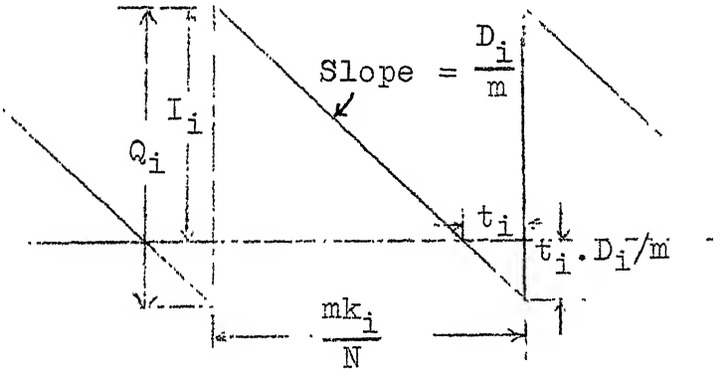


Fig. 3: Inventory Process for Problem 2. (Infinite horizon).

If the i-th item is replenished at every k_i-th set-up, then there will be N/k_i set-ups per year for this item. Here, k_i must be a positive integer and is called as relative ordering frequency for the i-th item. The annual variable cost VC_i for the i-th item is given by:

$$VC_i = \frac{N}{k_i} s_i + \tfrac{1}{2} \left(Q_i - \frac{t_i . D_i}{m}\right) \left(\frac{mk_i}{N} - t_i\right) \frac{N}{k_i} . h_i$$

$$+ \tfrac{1}{2} t_i^2 \frac{D_i . N}{mk_i} b_i + R . w_i \left(Q_i - \frac{t_i . D_i}{m}\right) \qquad (40)$$

The total annual cost for the inventory system is given by:

$$TC = NS + \sum_{i=1}^{n} \frac{N}{k_i} s_i + \tfrac{1}{2} \sum_{i=1}^{n} \left(Q_i - \frac{t_i . D_i}{m}\right)\left(\frac{mk_i}{N} - t_i\right)$$

$$h_i \frac{N}{k_i} + \tfrac{1}{2} \sum_{i=1}^{n} t_i^2 \frac{D_i . N}{mk_i} b_i + R\left[\sum_{i=1}^{n} w_i \left(Q_i - \frac{t_i D_i}{m}\right) - W\right] \qquad (41)$$

where the order quantity, Q_i for item i is given by.

$$Q_i = \frac{D_i . k_i}{N} \qquad (42)$$

In the expression for VC_i, substitute the value of Q_i from equation (42) and equate the partial derivative of VC_i with respect to t_i to zero. This gives optimum value of t_i as:

$$t_i = \frac{Rk_i . w_i + m . k_i . h_i}{N(h_i + b_i)} \qquad (43)$$

Substituting this value of t_i in equations (40) and (41), and replacing Q_i by $D_i . k_i / N_i$, we get:

$$VC_i = \frac{Ns_i}{k_i} + \frac{D_i . m}{2N} \cdot \frac{h_i . b_i}{h_i + b_i} . k_i$$

$$+ R \frac{w_i . D_i . k_i \left(2m . b_i - R . w_i\right)}{2m . N \left(h_i + b_i\right)} \qquad (44)$$

and,

$$TC = NS + \sum_{i=1}^{n} \left[\frac{Ns_i}{k_i} + \frac{D_i.m}{2N} \cdot \frac{h_i.b_i}{h_i+b_i} k_i + \right.$$

$$\left. + R \frac{w_i.D_i.k_i\ (2m.b_i - Rw_i)}{2m\ N\ (h_i + b_i)} \right] - R.W \qquad (45)$$

4. Solution Procedure:

Heuristic Algorithm 2:

In the above problem, two policy decisions are to be taken: the total number of set-upts per year N and the number of set-ups for individual items N/k_i. For a given value of N, the term NS in equation (45) is constant. The value of k_i for the i-th item can then be found by differentiating equation (44) partially with respect to k_i and putting it equal to zero. This gives:

$$k_i = N \left[\frac{2s_i.m\ (h_i + b_i)}{D_i.m^2.h_i.b_i + R.w_i.D_i\ (2mb_i - R.w_i)} \right]^{\frac{1}{2}} \qquad (46)$$

or,

$$\frac{N}{k_i} = \left[\frac{D_i.m^2.h_i.b_i + R.w_i.D_i\ (2mb_i - R.w_i)}{2s_i.m\ (h_i + b_i)} \right]^{\frac{1}{2}} \qquad (47)$$

We have assumed that atleast one item will be ordered at each set-up. This means atleast one of the k_i's will be equal to 1. The procedure adopted here can now be explained

as follows: First find out the values of N/k_i for each item from equation (47). Since one of the k_i's should be 1, therefore, start with the first item and put the corresponding value of k_i equal to 1. This gives a particular value of N. From this value of N find out the values of k_i for all other items. Calculate the total cost of the system with these values of N and k_i's. Next for the second item put $k_i = 1$ and repeat the whole process. The above procedure is followed for all items and the one which gives minimum total cost is selected.

A step by step algorithm for the above procedure is given below:

Step 1: Use the expression (47) to find out the value of N/k_i for the i-th item. Let this value of N/k_i be called as N_i. Repeat this process to determine values of N_i for all items.

Step 2: Atleast one of the items is to be ordered at all the setups or in other words it can be said that atleast one of the k_i's should be one. Let it be for the i-th item i.e., $k_i = 1$. So the value of N (i.e. total number of set-ups) will be the same as N_i.

Step 3: If for any other item j $(j \neq i)$ the value of N_j is greater than or equal to the value of N calculated in

Step 2 then for that item put $k^c_{ji} = 1$ and go to Step 9, otherwise continue.

Step 4: Calculate the value of k_{ji} for those items only for which N_j is less than N_i. The value of k_{ji} for such an item can be calculated from the relationship given below:

$$k_{ji} = \frac{N_i}{N_j}$$

Step 5: If for any item j, the value of k_{ji} is an integer then put $k^c_{ji} = k_{ji}$ and go to Step 9, otherwise continue.

Step 6: If k_{ji} is a fraction then an integer value of it is to be taken which gives the minimum variable cost for the j-th item. In such a case k_{ji} can take either lower integer value, say k_{jil}, or the next higher integer value, say k_{ji2}. First take the lower integer value and calculate the variable cost, say VC_{jl} for the j-th item with the value of N calculated in Step 2. The value of VC_{jl} can be calculated by using the following equation:

$$VC_{jl} = \frac{N_i . s_j}{k_{jil}} + \frac{D_j . m}{2N_i} \cdot \frac{h_j . b_j}{h_j + b_j} \cdot k_{jil} + \frac{R . w_j . D_j . k_{jil} \ (2m . b_j - R . w_j)}{2m . N_i (h_j + b_j)}$$

Step 7: Take the next higher integer value, k_{ji2}, and again calculate the variable cost, say VC_{j2}, for the same item, using the following equation:

$$VC_{j2} = \frac{N_i . s_j}{k_{ji2}} + \frac{D_j . m}{2N_i} \cdot \frac{h_j . b_j}{h_j + b_j} \cdot k_{ji2} + \frac{R . w_j . D_j . k_{ji2} \ (2m\, b_j - R . w_j)}{2m . N_i \ (h_j + b_j)}$$

Step 8: Compare the values of VC_{j1} and VC_{j2}. If VC_{j1} is greater than or equal to VC_{j2} then put $k^c_{ji} = k^2_{ji}$ and go to Step 9, otherwise, put $k^c_{ji} = k'_{ji}$.

Step 9: Repeat Steps 3 through 8 for all the values of j (i.e., for the value of N calculated in Step 2 find out the values of k^c_{ji} for all the items).

Step 10: Calculate the total cost for the set of k^c_{ji}'s and N values calculated above. The total cost, TC_i, can be computed using the following equation:

$$TC_i = N_i S + \sum_{j=1}^{n} \left[\frac{N_i s_j}{k^c_{ji}} + \frac{D_j . m}{2N_i} \cdot \frac{h_j . b_j}{h_j + b_j} \cdot k^c_{ji} + \frac{R . w_j . D_j . k^c_{ji} \ (2mb_j - R . w_j)}{2m\, N_i \ (h_j + b_j)} \right] - R . W$$

Step 11: Repeat the Steps 3 through 10 for i = 1, 2, ..., n (i.e. putting $k_i = 1$ for the next item calculate the values of N and k_{ji}'s and corresponding total cost).

Step 12: Select that value of N and set of k^c_{ji}'s for which the total cost TC_i is minimum. Let the selected values of k^c_{ji}'s be represented as k^*_i's for all items.

Step 13: Calculate the economic order quantity for all the items using the following expression:

$$Q^*_i = \frac{k^*_i \cdot D_i}{N}$$

Heuristic Procedure 3:

The method described in this section is an iterative procedure. To start with, a set of values of k_i are arbitrarily selected and the value of N is determined. The corresponding value of the objective function is then calculated. The value of the objective function is improved at each iteration. If for any two consecutive iterations, value of the objective function remains the same, then the procedure is stopped. The logic behind the proposed procedure is as follows:

For each set of values of k_i's there exists a local minimum. The value of N which corresponds to this minimum can be obtained by taking partial derivative of TC, in expression (45) with respect to N and equating it to zero. This gives:

$$N = \left[\frac{m^2 \sum_{i=1}^{n} D_i \cdot \frac{h_i \cdot b_i}{h_i + b_i} \cdot k_i + R \sum_{i=1}^{n} \frac{D_i \cdot w_i \cdot k_i (2mb_i - R \cdot w_i)}{(h_i + b_i)}}{2m \left(S + \sum_{i=1}^{n} \frac{s_i}{k_i} \right)} \right]^{\frac{1}{2}} \qquad (48)$$

Again, a local minimum exists for every value of N and corresponding values of k_i's for all items. For a given value of N, the term NS in the expression for total cost is a constant. The optimal value of k_i for item i can, therefore, be determined by taking the partial derivative of variable cost, VC_i, with respect to k_i and equating it to zero. This gives:

$$k_i = N \left[\frac{2s_i \cdot m \ (h_i + b_i)}{D_i \cdot m^2 \ h_i \cdot b_i + R \cdot w_i \cdot D_i \ (2mb_i - R \cdot w_i)} \right]^{\frac{1}{2}}$$

The economic number of set-ups, N, and relative ordering frequency k_i for the i-th item can be determined with the help of expressions (48) and (49). The value of N depends on a prior knowledge of k_i for each item, which in turn depends on N. An iterative procedure is, therefore, adopted for determining N and k_i for individual items.

For the r-th iteration in the iterative procedure, we

(a) use the $k_{i,r-1}$ values obtained in (r-1)-th iteration when N was equal to N_{r-1}, and

(b) Obtain N_r and $k_{i,r}$ for each item. If the value of $k_{i,r}$ is not an integer, then take the lower integer as value of $k_{i,r}$ and calculate the value of variable cost for i-th item. Let it be $VC_{i,1}$. Next, take the higher integer as value of $k_{i,r}$ and calculate the variable cost. Let it be $VC_{i,2}$. If $VC_{i,1}$ is less than $VC_{i,2}$ then the lower integer is taken as the value of $k_{i,r}$, otherwise, the higher integer is taken as $k_{i,r}$.

The iterative procedure will be terminated in r-th iteration if $k_{i,r} = k_{i,r-1}$ for all items.

The following steps are involved in the proposed iterative procedure:

Step 1: For the first iteration put $r = 1$ and select arbitrary values of k_{ir} for all items. In practice, it is found that values of k_{ir} for $i = 1,2,\ldots n$, provide a faster convergence. (If the arbitrarily selected values of k_{ir}'s happen to be near the optimal values of k_{ir}'s then the convergence with these values of k_{ir}'s will be faster.)

Step 2: Calculate the total number of set-ups N from the formula given below. Let it be represented as N_r. Thus, we get:

$$N_r = \left[\frac{m^2 \sum_{i=1}^{n} L_i \cdot \frac{h_i \cdot b_i}{h_i + b_i} \cdot k_{ir} + R \sum_{i=1}^{n} \frac{D_i \cdot w_i \cdot k_{ir} \, (2mb_i - R \cdot w_i)}{(h_i + b_i)}}{2m \left(S + \sum_{i=1}^{n} \frac{s_i}{k_{ir}} \right)} \right]^{\frac{1}{2}} \quad (50)$$

Step 3: Set r = r+1 and i = 0.

Step 4: Set i = i+1. Use the value of N_r calculated in Step 2 (i.e. in the previous iteration) to find out the new value of k_{ir}. The value of k_{ir} can be calculated from the formula given below:

$$k_{ir} = N_{r-1} \left[\frac{2 \, s_i \, m \left(\frac{h_i}{m} + b_i \right)}{D_i \cdot m^2 \cdot h_i \cdot b_i + R \cdot w_i \cdot D_i (2mb_i - R \cdot w_i)} \right]^{\frac{1}{2}} \quad (51)$$

Step 5: If the value of k_{ir} calculated in Step 4 is less than or equal to 1, then put $k_{ir}^c = 1$ (This is done because no set-up can be taken at intermediate points) and go to Step 10. If the value of k_i is greater than 1 then go to Step 6.

Step 6: If k_{ir}^c is an integer number then put $k_{ir}^c = k_{ir}$ and go to Step 10, otherwise, continue.

Step 7: Take the lower integer as value of k_{ir}. Let it be represented as k'_{ir}. Calculate the variable cost for i-th item from the formula given below. Let it be VC_{il}. Then VC_{il} is given by:

$$VC_{il} = \frac{N_{r-1} \cdot s_i}{k'_{ir}} + \frac{D_i \cdot m}{2N_{r-1}} \cdot \frac{h_i \cdot b_i}{h_i + b_i} \cdot k'_{ir}$$

$$\frac{R \cdot w_i \cdot D_i \cdot k'_{ir} \ (2mb_i - R \cdot w_i)}{2m \cdot N_{r-1} \ (h_i + b_i)}$$

Step 8: Take the next higher integer as value of k_{ir}. Let it be represented as k''_{ir}. Calculate the variable cost with this value of k_{ir}. Let it be VC_{i2}. Then VC_{i2} is given by:

$$VC_{i2} = \frac{N_{r-1} \cdot s_i}{k''_{ir}} + \frac{D_i \cdot m}{2N_{r-1}} \cdot \frac{h_i \cdot b_i}{h_i + b_i} \cdot k''_{ir}$$

$$+ \frac{R \cdot w_i \cdot D_i \cdot k''_{ir} \ (2mb_i - R \cdot w_i)}{2m \cdot N_{r-1} \ (h_i + b_i)}$$

Step 9: Compare the value of VC_{il} with VC_{i2}. If VC_{il} is less than VC_{i2}, then put $k^c_{ir} = k'_{ir}$ and go to Step 10, otherwise, put $k^c_{ir} = k''_{ir}$.

Step 10: If the value of i isless than total number of items n, then go to Step 4, otherwise, continue.

Step 11: If $k^c_{ir} = k_{i,r-1}$ for i = 1,2,..., n (i.e. if the two consecutive values of k_{ir} are same for all the items) then go to Step 12, otherwise, put $k_{ir} = k^c_{ir}$ for i = 1,2,..., n and go to Step 2.

Step 12: The economic number of set-ups per year N is equal to N_{r-1}. Find out the economic order quantity for each item, using the following formula:

$$Q_i^* = \frac{D_i \cdot k_{ir}}{N^*}$$

Step 13: Determine the number of set-ups for each item from the following formula:

$$N_i^* = \frac{N}{k_{ir}}$$

Step 14: Substitute the values of N and k_{ir} for $i = 1,2,\ldots,n$ into equation (45) and calculate the total cost of the system.

CHAPTER IV

METHODOLOGY TESTING AND SCOPE FOR FUTURE WORK

In this chapter results pertaining to the performance of the proposed models on a few test problems are presented. At the end of the chapter, possible avenues for further work are suggested.

4.1 Performance of Dynamic Programming Procedure:

Nine problems have been solved using discrete dynamic programming computer package and for each problem optimum ordering policies have been determined. The problem size is varied from two items to four items. For each problem size a set of three problems have been solved. The average computation time for these problems on the IBM 7044 Computer system are given in Table 4.1. For the sake of illustration, the complete results of only one problem, for which the data is given in Table 4.2, are presented. For this problem, the total available warehouse space is taken as 16 units. The results for the optimum ordering policies (i.e., procurement quantity and procurement level for each item) are given in Table 4.3.

The same problem has also been solved using continuous dynamic programming procedure. The computer package by Mize [14] was utilized for this purpose. The optimum

ordering policies for the problem, as obtained by the continuous dynamic programming procedure are given in Table 4.4.

It is found that the discrete dynamic programming procedure takes a computational time of 2.2 seconds and gives an objective function valueof 284.689 units. The continuous dynamic programming results in a better objective function value of 278.736 units but it takes 17.605 seconds of computational time. Therefore, it is inferred that continuous dynamic programming approach yields better results, however, at a higher computational time.

4.2 Performances of Heuristic Algorithms:

The performance of the three heuristics discussed in Chapter 3 have been compared by solving about 150 problems of various sizes. The problem size is varied from 6 items to 40 items. For each problem size a set of problems have been generated randomly. Computer packages have been developed in FORTRAN IV for the IBM 7044/1401 Computer system, for testing the proposed heuristics.

The performances of the heuristics are evaluated based on cost index and average computational time for a given problem size. The details of the two measures of evaluation are given in the following paragraphs.

Cost Index:

For a given problem size, n, let TC_{en} and TC_{in} be the total system cost using economic order quantity model and heuristic approach 'i' respectively. Then the cost index, CI_{in}, is determined using the following relationship:

$$CI_{in} = \frac{TC_{in}}{TC_{en}}$$

where, i = index of heuristic (i = 1,2,3)

n = index of problem size (n = 6,7,..., 40)

For a given problem the heuristic which gives least value of the cost index is taken as the best heuristic.

The three heuristics have been compared for a set of 45 problems chosen randomly. Heuristic one gives the total system cost for a given planning horizon, whereas, heuristics two and three give annual total system cost. To compare heuristic one with heuristics two and three, the planning horizon for heuristic one is taken as one year. The results are presented in Table 4.5. The results indicate that heuristic three yields least cost index values in most of the cases (42 out of 45). In three cases the cost index values obtained by heuristic two are least and in 11 cases these are very near to those obtained by heuristic three. The higher cost index values for heuristic one can be attributed to the following reasons which accrue from the

two constraints involved in this heuristic:

1. There is an upper limit on the total number of set-ups per year,
2. The total number of inventory cycles for each item has to be an integer.

Further, examination of Table 4.5 indicates that the cost index values obtained for each of the heuristics is less than 1. This implies that it is economical to replenish several items jointly rather than ordering them separately.

Average Computational Time:

For all the generated problems, the average computational time is determined for each problem size. Average computational time values for different problem sizes are presented in Table 4.6. It can be seen from the table that for small size problems (upto 7 items) heuristic two takes the least computational time. Heuristic three closely follows heuristic two. For large size problems (from 8 items to 40 items) heuristic three takes least time. A possible explanation for this behaviour of heuristics is given below:

Heuristic two generates multiple solutions (equal to the number of items) for a given problem. Out of these solutions, the minimum objective function value solution is selected. Thus the computational time of heuristic two

increases as the problem size increases. Heuristic three, on the other hand, is an iterative procedure, and as will be shown latter, the number of iterations do not depend on the problem size (the number of iterations, in general, are between 6 to 10). Therefore, for large size problems heuristic three takes much less time compared to heuristic two.

Based on the two measures of evaluation, it can be concluded that heuristic three is the most efficient heuristic.

Heuristic three, being an iterative procedure, is further tested to evaluate for its convergence. This is done on the basis of number of iterations it takes to solve a problem of given size and the percentage improvement in the total system cost.

For a problem of size 'n', let TC_{1n} and TC_{Ln} represent the total system cost at the first iteration and at the last iteration respectively. Then the percentage improvement, PI_n, is given by the following relationship:

$$PI_n = \frac{TC_{1n} - TC_{Ln}}{TC_{1n}} \times 100$$

Table 4.7 gives the values of PI_n for 132 problems ranging in size from 6 items to 40 items. The second column of the table gives the number of problems solved

I.I.T. KANPUR
CENTRAL LIBRARY
Acc. No. 46897

for the problem size under consideration. The third and fourth column of the table give the minimum andmaximum values of the number of iterations and percentage improvements respectively. As can be seen from the table, the number of iterations in most of the cases are less than 10 (only in 4 cases the number of iterations exceeded 10). Thus, it can be concluded that number of iterations do not depend on the problem size. Moreover, considerable improvement in the objective function value can be made in a few iterations. To illustrate it, for an 11 item problem percentage improvement of 88.13 is obtained in 9 iterations.

In the previous paragraphs we have compared the three heuristics when the planning horizon is infinite. Now we shall evaluate the performance of heuristic one which considers finite planning horizon. The evaluation of this heuristic is done using the following procedure.

The total system cost is calculated using the heuristic. The total system cost is also calculated when all the items are ordered at each period. The total system cost values using both the procedures are compared.

For a problem of size, n, let TC_{An} represent the total system cost when all the items are ordered at each period and let TC_{1n} represent the total system cost calculated by heuristic one. The percentage cost saving, PS_n,

obtained by using heuristic one is defined as:

$$PS_n = \frac{TC_{An} - TC_{1n}}{TC_{An}} \times 100$$

where n = index for problem size.

Table 4.8 gives the values of PS_n for 131 problems for the problem sizes ranging from 6 items to 40 items. From the table, it is clear that the use of heuristic one as compared to the procedure of ordering each item at every period in the planning horizon results in a saving. For the selected set of problems, savings upto 97.08 percent were obtained by the use of heuristic one.

4.3 Scope for Future Work:

In developing the methodologies presented in this dissertation, a few avenues for further work emerged. These are given below:

1. In the dynamic programming procedure used it is assumed that the demand and lead time are deterministic. It will be worthwhile to extend it to the case where demand and/or lead time are stochastic.

2. In heuristics two and three it is assumed that there is no limit on the total number of set-ups per year. It would be interesting to analyse the problem when there is an upper limit on the total number of set-ups in a year.

3. In the three heuristic procedures developed in this work, it is assumed that product interactions only due to joint replenishment of the products are present. It would be worthwhile to consider the cases where additional product interactions due to warehouse capacity restriction and/or budgetary restriction are also present.

Table 4.1: Average Computational Time for Problems Solved by Discrete Dynamic Programming Procedure

S.No.	Problem Size	Average Computational Time (Sec.)
1	2	2.333
2	3	3.075
3	4	3.992

Table 4.2: Data Used for a Three Item Inventory Problem Solved by Both Discrete and Continuous Dynamic Programming Procedures.

Item No.	Demand Rate	Item Cost	Holding Cost	Backorder Cost	Procurement Cost	Lead Time	Space consumed by one unit
1	6	30.88	0.30	0.30	18.30	2	4
2	4	18.33	0.24	0.17	17.50	4	2
3	1	12.00	0.12	0.25	15.50	1	1

Table 4.3: Optimal Ordering Policies Obtained by Discrete Dynamic Programming Procedure for the Problem Given in Table 4.2

Item No.	Procure-ment Level	Procurement Quantity	Warehouse space allotted
1	-14.08	27.10	4
2	-10.06	29.07	6
3	- 6.31	13.31	6

Total Cost: 284.689 units

Computer Time Taken: 2.2 seconds

Table 4.4: Optimal Ordering Policies Obtained by Continuous Dynamic Programming Procedure for the Problem given in Table 4.2.

Item No.	Procurement Level	Procurement Quantity	Warehouse space allotted
1	-12.91	27.26	9.400
2	-11.20	28.80	3.200
3	- 8.80	11.28	1.472

Total Cost: 278.736 units

Computer Time Taken: 17.605 seconds

Table 4.5: Performance Comparison of Three Heuristics Based on Cost Index Values.

Sl. No.	Problem Size	Cost Index Values		
		Heuristic 1	Heuristic 2	Heuristic 3
1	6	0.482	0.447	0.331
2	7	0.637	0.610	0.509
3	7	0.904	0.889	0.887
4	8	0.704	0.636	0.634
5	9	0.723	0.637	0.636
6	10	0.309	0.288	0.202
7	11	0.482	0.477	0.420
8	12	0.485	0.478	0.421
9	13	0.421	0.320	0.249
10	14	0.246	0.230	0.216
11	15	0.320	0.294	0.217
12	16	0.591	0.594	0.592
13	16	0.960	0.968	0.944
14	17	0.211	0.187	0.163
15	17	0.318	0.324	0.283
16	18	0.278	0.249	0.151
17	19	0.591	0.580	0.555
18	19	0.982	0.953	0.922
19	20	0.527	0.527	0.486
20	21	0.374	0.363	0.350
21	22	0.723	0.538	0.487
22	23	0.641	0.623	0.623
23	24	0.623	0.549	0.556
24	25	0.583	0.548	0.554
25	26	0.712	0.698	0.488
26	27	0.281	0.299	0.302
27	28	0.136	0.117	0.115
28	29	0.163	0.146	0.137
29	30	0.124	0.114	0.112
30	31	0.143	0.149	0.129
31	32	0.208	0.187	0.103
32	33	0.189	0.153	0.153
33	34	0.168	0.131	0.118
34	35	0.178	0.246	0.135
35	35	0.507	0.532	0.482
36	36	0.126	0.116	0.112
37	36	0.667	0.675	0.632
38	37	0.292	0.284	0.169
39	37	0.862	0.831	0.782
40	38	0.224	0.164	0.103
41	38	0.703	0.675	0.621
42	39	0.152	0.109	0.099
43	39	0.578	0.527	0.483
44	40	0.173	0.162	0.160
45	40	0.806	0.713	0.629

Table 4.6: Performance Evaluation of the Three Heuristics Based on Average Computational Time.

Sl. No.	Problem Size	Average Computational Time (Sec.)		
		Heuristic 1	Heuristic 2	Heuristic 3
1	6	0.378	0.267	0.270
2	7	0.420	0.291	0.295
3	8	0.480	0.368	0.342
4	9	0.503	0.434	0.320
5	10	0.984	0.483	0.429
6	11	0.729	0.617	0.392
7	12	0.820	0.723	0.515
8	13	0.793	0.936	0.581
9	14	0.840	0.927	0.601
10	15	1.082	1.038	0.642
11	16	0.960	1.026	0.687
12	17	1.020	1.015	0.730
13	18	1.082	1.061	0.720
14	19	1.146	1.127	0.761
15	20	1.208	1.201	0.804
16	21	1.403	1.333	0.785
17	22	1.308	1.293	0.770
18	23	1.382	1.321	0.920
19	24	1.458	1.431	0.963
20	25	1.489	1.471	0.916
21	26	1.621	1.599	0.923
22	27	1.672	1.660	0.978
23	28	1.998	2.016	1.023
24	29	2.123	2.088	1.042
25	30	2.004	1.950	0.857
26	31	2.087	2.021	0.885
27	32	2.135	2.080	0.913
28	33	2.263	2.145	0.947
29	34	2.371	2.212	0.963
30	35	3.398	3.375	1.021
31	36	2.367	2.341	1.028
32	37	2.428	2.417	1.056
33	38	2.491	2.483	1.073
34	39	2.571	2.568	1.110
35	40	2.648	2.630	1.138

Table 4.7: Convergence Evaluation of Heuristic 3.

Sl. No.	Problem Size	No. of Problems Solved	Range of Total No. of Iterations	Range of Percentage Improvements
11	6	3	6 - 7	28.34 - 65.34
2	7	3	4 - 10	22.84 - 38.85
3	8	4	4 - 8	9.73 - 59.83
4	9	4	7 - 10	55.27 - 83.97
5	10	4	6 - 8	60.44 - 64.28
6	11	4	5 - 9	34.00 - 88.13
7	12	5	6 - 8	48.64 - 58.13
8	13	4	6 - 9	77.77 - 80.23
9	14	4	6 - 10	64.66 - 89.21
10	15	4	6 - 12	46.67 - 65.15
11	16	3	7 - 9	38.84 - 57.36
12	17	4	5 - 8	43.86 - 68.94
13	18	5	6 - 9	35.73 - 58.75
14	19	3	8 - 9	28.15 - 64.42
15	20	4	7 - 9	20.15 - 52.34
16	21	4	9 - 12	52.54 - 71.67
17	22	3	7 - 10	32.73 - 55.25
18	23	4	7 - 14	58.63 - 84.58
19	24	4	6 - 8	26.98 - 63.72
20	25	4	6 - 14	27.06 - 85.58
21	26	3	6 - 10	52.93 - 88.90
22	27	5	6 - 9	54.79 - 88.63
23	28	4	6 - 8	29.36 - 89.06
24	29	3	5 - 10	26.63 - 86.31
25	30	4	6 - 8	29.11 - 62.46
26	31	4	5 - 9	26.38 - 64.76
27	32	4	5 - 8	9.22 - 35.27
28	33	4	7 - 10	42.82 - 64.17
29	34	4	5 - 8	28.65 - 68.48
30	35	3	6 - 9	33.32 - 47.46
31	36	3	9 - 10	37.40 - 53.58
32	37	3	8 - 8	47.34 - 58.93
33	38	4	9 - 9	50.05 - 69.28
34	39	4	7 - 9	51.04 - 73.82
35	40	4	6 - 8	42.92 - 81.34

Table 4.8: Range of Percentage Cost Savings Obtained by Heuristic One as Compared to the Procedure in which All the Items are Ordered at Each Set-up.

(Total Problem = 131)

Sl. No.	Problem Size	Number of Problems Solved	Range of Percentage Cost Savings
1	6	5	52.79 - 94.84
2	7	5	9.88 - 85.52
3	8	4	22.31 - 85.27
4	9	4	57.32 - 75.53
5	10	5	28.72 - 94.76
6	11	4	11.72 - 72.87
7	12	4	19.94 - 76.93
8	13	5	38.32 - 84.50
9	14	5	16.99 - 87.73
10	15	3	58.20 - 82.47
11	16	4	67.37 - 94.08
12	17	4	68.62 - 85.68
13	18	4	32.64 - 87.50
14	19	5	77.70 - 92.08
15	20	3	82.33 - 85.94
16	21	4	81.27 - 82.82
17	22	4	48.82 - 79.03
18	23	3	32.74 - 44.25
19	24	3	28.35 - 60.26
20	25	5	40.82 - 62.35
21	26	3	70.76 - 97.08
22	27	4	44.26 - 76.56
23	28	4	21.79 - 53.74
24	29	4	52.35 - 74.86
25	30	3	21.59 - 42.73
26	31	3	20.95 - 68.74
27	32	3	4.05 - 29.87
28	33	3	43.68 - 68.39
29	34	3	22.15 - 24.12
30	35	3	29.22 - 52.19
31	36	4	18.04 - 19.84
32	37	3	31.48 - 33.61
33	38	4	28.03 - 31.17
34	39	3	34.06 - 37.90
35	40	3	31.96 - 64.50

REFERENCES

1. Andrews, F., and Emmons, H., 'A Multiproduct Inventory System with Interactive Set-up Costs', Management Science, Vol. 21, No. 9, 1975, pages 1055-1063.

2. Arrow, K.J., T.Harris, and J. Marsehak, 'Optimal Inventory Policy', Econometrica, Volume 14, 1957, pages 250-272.

3. Buchan, J., and Koenigsberg, E., 'Scientific Inventory Management', Prentice-Hall, 1963.

4. Davis, R., 'Optimal Inventory Control Decision Rules for a Large Supply System', Operations Research, Volume 7, No. 6, Nov. Dec. 1959, pages 764-782.

5. Evans, R.V., 'Inventory Control of a Multi-Product System with Limited Production Resource', Naval Research Logistics Quarterly, 1967, pages 173-184.

6. Hadley, G., and Whitin, T.M., Analysis of Inventory Systems, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1963.

7. Hanssmann, F., Operations Research in Production and Inventory Control, John Wiley and Sons, 1962.

8. Herron, D.P., 'Inventory Management for Minimum Cost', Management Science, Volume 14, No. 4, 1967, pages B 219-B 235.

9. Herron, D.P., and R.L. Hawley, 'Establishing an Optimum Inventory Size and Stocking Policy for a Warehouse', AIIE Transactions, Volume 1, No. 1, March, 1969, pages 75-80.

10. Hoffman, T.R., Production: Management and Manufacturing Systems, Prentice Hall, 1967.

11. Ignall, E., 'Optimal Continuous Review Policies for two Product Inventory Systems with Joint Set-up Costs', Management Science, Vol. 15, No. 5, pages 278-283.

12. Inglehart, D.L., 'A Dynamic Inventory Problem with Unknown Demand Distribution', Management Science, Vol. 10, No. 3, 1964, pages 429-440.

13. Karlo, A.H., 'Comments on 'Economic Ordering Frequency for Two Items Jointly Replenished' by F.T. Shu', Management Science, Vol. 18, No.12, 1972, pages B 732-B 733.

14. Kuester, J.L., and Mize, J.H., 'Optimization Techniques with Fortran', McGraw-Hill Book Company, 1973.

15. Larson, R.E., State Increment Dynamic Programming, New York, American Elsevier Publishing Co., 1968.

16. Lele, A.R., 'An Approach to Multi-Product Inventory Control Under Budgetary and Storage Restrictions', M.Tech. Thesis Submitted to IIT Kanpur, August 1973.

17. Naddor, E., Inventory Systems, New York, Wiley, 1966.

18. Naddor, E., and Saltzman, 'Optimal Reorder Periods for an Inventory System with Variable Costs of Ordering', Operations Research, Vol. 6, No.5, 1958.

19. Newbery, T.L., 'Classification of Inventory Control Theory', Journal of Industrial Engineering, Vol. 11, No. 5, 1960, pages 391-397.

20. Plossl, G.W., Harty, J.D., and Wight, O.W., 'Management of Lot Size Inventories', A Special Education and Research Report of the American Production and Inventory Control Society, Chicago, Illinois, 1963.

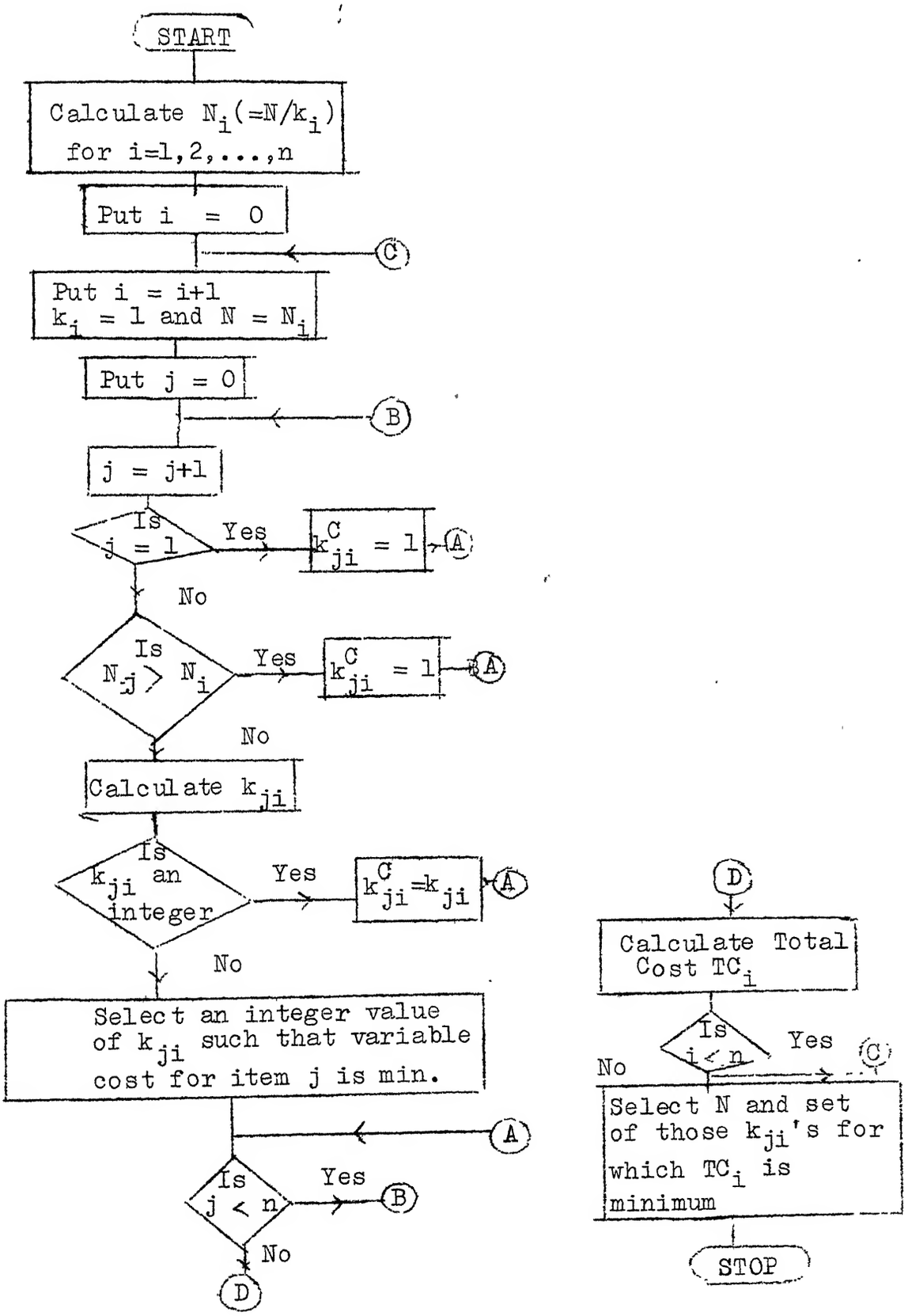
21. Saxton, G.F., 'An Approach to Evaluate Interactions in a Multi-Product Total Cost Inventory Model', Ph.D. Thesis Submitted to Arizona State University, Feb., 1972 (Photostat Copy).

22. Shu, F.T., 'Economic Ordering Frequency for Two Items Jointly Replenished', Management Science, Vol. 17, No. 5, 1971, pages B 406 - B 410.

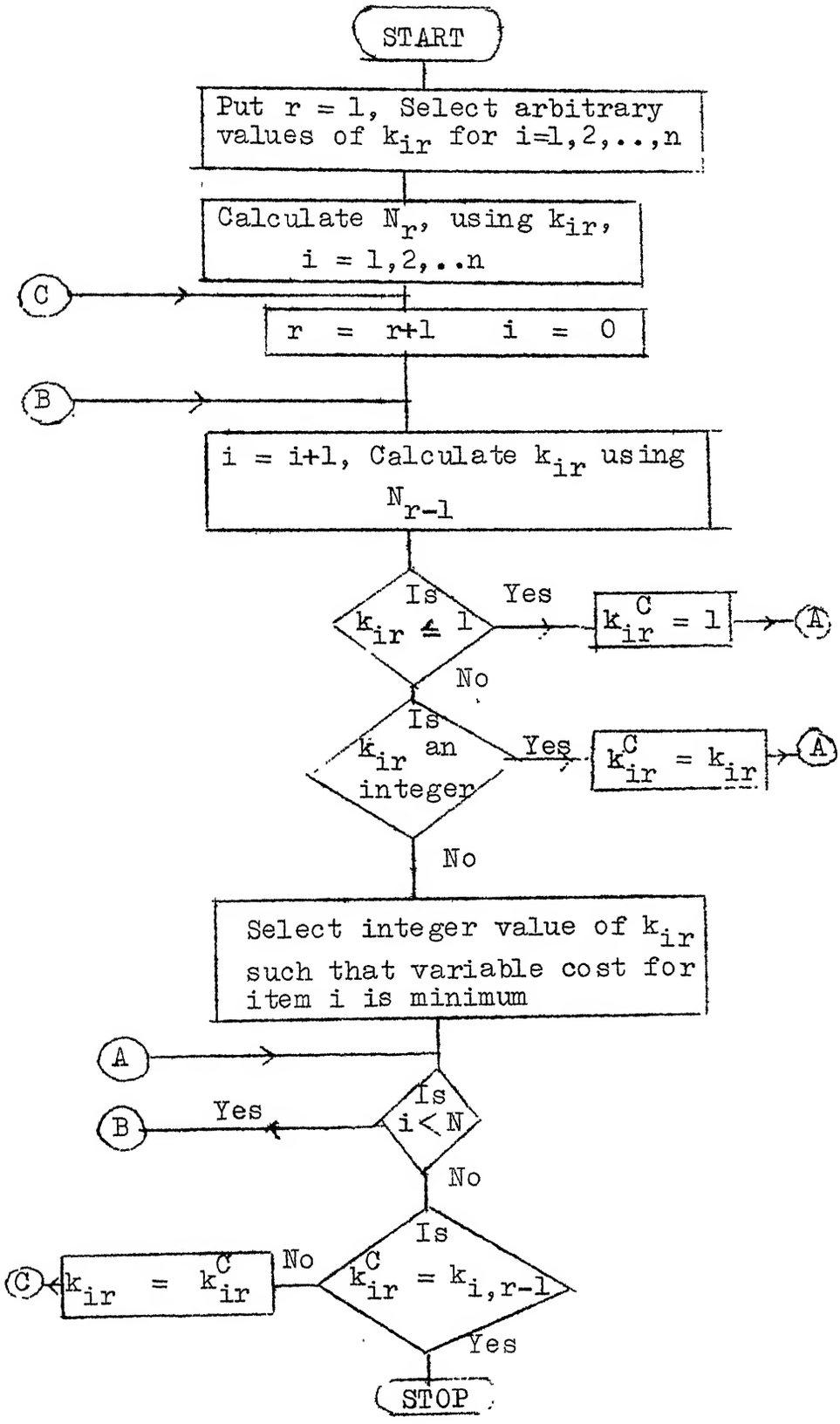
23. Starr, M.K., and Miller, D.W., Inventory Control: Theory and Practice, Prentice-Hall, 1962.

24. Wagner, H.M., 'Principles of Operations Research with Applications to Managerial Decisions', Prentice Hall of India Pvt. Ltd., 1974.

FLOW CHART FOR HEURISTIC TWO



FLOW CHART FOR HEURISTIC THREE



```
C
C*****************************************************************************
C
C       COMPUTER PROGRAM TO TEST THE PERFORMANCE OF HEURISTIC
C
C*****************************************************************************
C
C       D(I) = DEMAND RATE FOR ITEM I
C       S = FIXED SET UP COST
C       VS(I) = VARIABLE SET UP COST FOR ITEM I
C       H(I) = HOLDING COST FOR ITEM I
C       B(I) = BACKORDER COST FOR ITEM I
C       W(I) = WAREHOUSE SPACE OCCUPIED BY ONE UNIT OF ITEM I
C       Q(I) = ECONOMIC ORDER QUANTITY FOR ITEM I
C       R = TOTAL AVAILABLE WAREHOUSE SPACE
C       PRV = PERCENTAGE SAVING
C
C*****************************************************************************
C
      DIMENSION D(  ),VS(  ),H(  ),B(  ),IFRAC(  ),IN(  ),FRAC(  ),
     NDK(  ),NP(  ),NMP(5 ),VC(  ,  ),IM(5 ),IN(  ,5 ),IP(  ),IT(  )
     ),Q(  ),RKK(  ),RKM(5 ),  (  )
C
C*****************************************************************************
C
      READ  ,NPROP
      DO  (  )
      KOUNT =
      KOUNT = KOUNT +
      READ  ,M,NITEM
      N = NITEM
      PRINT  5,N
    1   FORMAT (/, X,*A*,I2,*ITEM INVENTORY PROBLEM*)
      READ  ,S,R,P,
      FORMAT (  /)
      DO  I = 1,N
      READ  ,D(I),V(I),H(I),B(I)
      FORMAT (  )
      CONTINUE
      PRINT  2
    2   FORMAT (/,*X,*ANSWER TO THE ABOVE PROBLEM IS GIVEN BELOW*)
C
C       ECONOMIC NO. OF SET UPS FOR EACH ITEM
C
    9 RM = M
      IFRAC( ) =
      II =
      K =
      II = II+
      II = II
      R = RM/P
      RN = M/II
      IF (RN - N)   ,   ,
      K=K+
```

```
      IFRAC(K) = 1I
      IF (I-M) [illegible]
      DO [illegible] I= ,N
        (I)=((1**[illegible])*D(I)*H(I)*B(I)+R*W(I)*D(I)*([illegible]*T*B(I)-W(I)*R))/
     [illegible]*VS(I)*(H(I)+B(I)))
      IN(I) = SQRT(IN(I))
      CONTINUE
      IJ =
      IJ = IJ+
      KK =
      KK = KK+
      FRAC(KK) = IFRAC(KK)
      IF ( IN(IJ) .GT. [illegible]) GO TO [illegible]
      NPK(IJ) =
      GO TO [illegible]
      IF ( IN(IJ) .EQ. FRAC(KK)) GO TO [illegible]
      IF ( IN(IJ) .GT. FRAC(KK)) GO TO [illegible]
      GO TO 8
    4 IF (KK-K) [illegible]
      NPK(IJ) = M
      GO TO [illegible]
    7 NPK(IJ) = IFRAC(KK)
      GO TO [illegible]
      NP(IJ) = FRAC(KK)
      VC(IJ,[illegible]) = NP(IJ)*VS(IJ)+([illegible]**2)*D(IJ)*H(IJ)*B(IJ)/([illegible]*NP(IJ)*
     (H(IJ) + B(IJ)))+R*W(IJ)*D(IJ)*([illegible]*T*B(IJ)-W(IJ)*R)/([illegible]*NP(IJ)
     *(H(IJ)+B(IJ)))
      KK = KK
      MP(IJ) = FRAC(KK)
      VC(IJ,[illegible]) = MP(IJ)*VS(IJ)+([illegible]**2)*D(IJ)*H(IJ)*B(IJ)/([illegible]*MP(IJ)*
     (H(IJ) + B(IJ)))+R*W(IJ)*D(IJ)*([illegible]*T*B(IJ)-W(IJ)*R)/([illegible]*MP(IJ)
     *(H(IJ)+B(IJ)))
      IF (VC(IJ,[illegible]) - VC(IJ,[illegible])) 9 , [illegible],
    9 NPK(IJ) = NP(IJ)
      GO TO [illegible]
      NPK(IJ) = MP(IJ)
      IF (IJ-N) 4 , [illegible] ,
C
C     THIS PART OF THE PROGRAM CALCULATES THE TOTAL
C     [illegible] OUTPUTS, TOTAL SYSTEM COST AND ECONOMIC ORDER
C     FOR ALL ITEMS
C
      DO [illegible] I= ,N
      IM(I) = M/NPK(I)
      NPKI = NPK(I)
      DO [illegible] J = [illegible],NPK
      IN(I,J) = [illegible]+(J- )*IM(I)
    [illegible] CONTINUE
      KR =
      DO [illegible] I = [illegible],N
      NPKI = NPK(I)
      JK =
    7 JK = JK+
      KR = KR +
```

```
IP(KR) = IN(I,JK)
IF (JK-NPKI) [illegible] 4, 4
CONTINUE
IP =
IP(ITP) = IP( )
DO [illegible] JP = ,KR
DO [illegible] KP = ,ITP
IF (IP(JP) [illegible] Q. IT(KP)) GO TO [illegible]
CONTINUE
ITP = ITP +
IT(ITP) = IP(JP)
CONTINUE
PRINT [illegible] ITP
    FORMAT ( [illegible],*TOTAL NUMBER OF SET UPS = *,I4)
PRINT [illegible]
    FORMAT ( [illegible],*THE SET UPS SHOULD BE TAKEN UP AT THE FOLLOWING P R
IODS *)
PRINT [illegible],(IT(KT),KT = ,ITP)
FORMAT ( I4)
COST =
DO [illegible] I= ,N
RKK(I) = NPK(I)
COST = COST +RKK(I)*VS(I)+(T** )*D(I)*H(I)*B(I)/([illegible]*RKK(I)*(H(I)+
B(I)))+R*W(I)*D(I)*(2 *T*B(I)-W(I)*R)/([illegible]*RKK(I)*(H(I)+B(I)))
CONTINUE
PCOST = ITP
PCOST = ITP*S + COST -R*P
PRINT [illegible]
FORMAT ( [illegible],* ITEM NO.    ORDER QUANTITY    SET UPS SHOULD BE TAKEN
 FOLLOWING P PIODS*)
DO [illegible] I= ,N
Q( ) = [illegible]*D(I)/RKK(I)
MPKI = NPK(I)
IF (MPKI .GT. [illegible]) GO TO [illegible]
PRINT [illegible],I,Q(I),(IN(I,J), J = ,MPKI)
FORMAT ( [illegible],I [illegible] 2.6,[illegible],2 I4)
GO TO [illegible]
PRINT [illegible] ,I,Q( ),(IN(I,J), J=[illegible],[illegible])
FORMAT ( [illegible],I [illegible] 2.6,[illegible], I4)
PRINT [illegible],(IN(I,J), J= [illegible],MPKI)
FORMAT ( [illegible], I [illegible])
CONTINUE
PRINT [illegible],TCOST
    FORMAT ( [illegible],*TOTAL COST OF THE SYSTEM = *,[illegible] 6.8)
COST =
DO [illegible] = ,N
RKM(I) = M
COST = COST +RKM(I)*VS(I)+(T**[illegible])*D(I)*H(I)*B(I)/([illegible]*RKM(I)*(H(I)+
B(I)))+R*W(I)*D(I)*([illegible]*T*B(I)-W(I)*R)/([illegible]*RKM(I)*(H(I)+B(I)))
CONTINUE
PCOST = RM*S + COST - R*P
PSIV =((PCOST - TCOST)/PCOST)*[illegible]
PRINT [illegible],PCOST
FORMAT ( [illegible],*TOTAL COST WHEN ALL ITEMS ARE ORDERED AT EACH SET UP
```

```
      = *(  .5)
      PRINT  ,PSAV
      FORM T(  ,*PERC NTAGE SAVING = *,  .8)
    9 IF (KCUNT .LT  .PRCB) GO TO 
      STOP
      END
```

```
C
C*****************************************************************************
C
C       COMPUTER PROGRAM TO TEST THE PERFORMANCE OF HEURISTIC 2
C
C*****************************************************************************
C
C       D(I) = DEMAND FOR ITEM I
C       S = FIXED SET UP COST
C       VS(I) = VARIABLE SET UP COST FOR ITEM I
C       H(I) = HOLDING COST FOR ITEM I
C       B(I) = BACKORDER COST FOR ITEM I
C       W(I) = WAREHOUSE SPACE OCCUPIED BY ONE UNIT OF ITEM I
C       Q(I) = ECONOMIC ORDER QUANTITY FOR ITEM I
C       P= TOTAL AVAILABLE WAREHOUSE SPACE
C
      DIMENSION VS(50),D(50),H(50),B(50),W(50),XSQ(50),X(50),EN(50),
     AK(50,50),NK(30,30),VC(30,30),MK(30,30),NPK(30,30),TC(40),BK(30,
     2),Q(50),RPK(30,30),PPK(30,30),ECOST(50)
      READ 1,NPROB
    1 FORMAT (8I10)
      KOUNT =
   10 KOUNT = KOUNT+1
      READ 2,N,NITEM
      M = NITEM
      PRINT 4,NITEM
    4    FORMAT (/,2X,*A*,I2,* ITEM INVENTORY PROBLEM SUBJECT TO WAREH
     OUSE CAPACITY CONSTRAINT *)
      READ 5,S,R,P
    5 FORMAT (3F10.4)
      PRINT 6,S
    6    FORMAT (2X,*FIXED SET UP COST = *,E13.6)
      PRINT 6 ,R
    6    FORMAT (2X,*WAREHOUSE RENT = *,E13.6)
      PRINT 7,P
    7    FORMAT (2X,*TOTAL AVAILABLE WAREHOUSE SPACE = *,E 3.6)
      PRINT 8,M
    8    FORMAT (2X,*NUMBER OF PERIODS/YEAR = *,I4)
      PRINT 9
    9    FORMAT (2X,*ITEMNO.  VARIABLE SET UP COST  YEARLY DEMAND  HOLDI
     NG COST  BACKORDER COST  SPACE OCCUPIED /UNIT*)
      DO    I = 1,N
      READ 3,D(I),VS(I),H(I),B(I),W(I)
    3 FORMAT (8F10.4)
      PRINT 10,I,VS(I),D(I),H(I),B(I),W(I)
   10    FORMAT (4X,I ,5X,E13.6,1X,E13.6,2X,E13.6,2X,E13.6,4X,E13.6)
      CONTINUE
      PRINT 12
   12    FORMAT (/,2X,*ANSWER TO THE ABOVE PROBLEM IS GIVEN BELOW*)
C
      DM = M
      DO 20 I = 1,N
      XSQ(I) = (D(I)*(DM**2)*H(I)*B(I)+R*D(I)*W(I)*(2.*
     1DM*B(I)-R*W(I)))/(2.*VS(I)*DM*(H(I)+B(I)))
```

```
      X(I) = SQRT(XSQ(I))
      CONTINUE
      DO 2  I = 1,N
      EN(I) = X(I)
      II = I
      DO 2  J = 1,N
      JJ = J
      IF (JJ.EQ.II) GO TO 27
      IF(EN(I) .LE. X(J)) GO TO 27
      NK(I,J) = EN(I)/X(J)
      AK(I,J) = NK(I,J)
      VC(J,1) = EN(I)*VS(J)/AK(I,J)+D(J)*CM*H(J)*B(J)*AK(I,J)/(2.*EN(I)*
     (H(J)+B(J)))+R*W(J)*AK(I,J)*D(J)*(2.*CM*B(J)-
     *R*W(J))/(2.*CM* N(J)*(H(J)+B(J)))
      MK(I,J) = NK(I,J) + 1
      BK(I,J) = MK(I,J)
      VC(J,2) = EN(I)*VS(J)/BK(I,J)+D(J)*CM*H(J)*B(J)*BK(I,J)/(2.*EN(I)*
     (H(J)+B(J)))+R*W(J)*BK(I,J)*D(J)*(2.*CM*B(J)-
     *R*W(J))/(2.*CM* N(J)*(H(J)+B(J)))
      IF (VC(J,1) -VC(J,2)) 37,47,47
   37 NPK(I,J) = NK(I,J)
      GO TO 51
   47 NPK(I,J) = MK(I,J)
      GO TO 51
   51 NPK(I,J) =
    2 CONTINUE
      C = 0.
      KK =
    7 KK = KK + 1
      DO 7  J = 1,N
      PPK(KK,J) = NPK(KK,J)
      C = C+EN(KK)*VS(J)/PPK(KK,J) + D(J)*CM*H(J)*B(J)*PPK(KK,J)/(2.*
     EN(KK)*(H(J) + B(J)))+ R*W(J)*PPK(KK,J)*D(J)*(2.*CM*B(J)
     2-R*W(J))/(2.*CM*EN(KK)*(H(J)+B(J)))
    7 CONTINUE
      TC(KK) = EN(KK)*S + C - R*P
      IF (KK - N) 7,67,67
   67 MN =
    7 MM = MN
      IF (MM .EQ. N) GO TO 87
      MN = MM + 1
   77 IF (TC(MM) .LE. TC(MN)) GO TO 57
      GO TO 77
   57 IF (MN .EQ. N) GO TO 87
      MN = MN +1
      GO TO 77
   87 DO 7  J = 1,N
      RPK(MM,J) = NPK(MM,J)
      Q(J) = D(J)*RPK(MM,J)/EN(MM)
    7 CONTINUE

      PRINT  2
    3    FORMAT (3X,*ITEM NO.    VALUE OF K    OPTIMUM ORDER QUANTITIES*)
      DO 8  J = 1,N
```

```
      PRINT [illegible] ,J,NPK(MM,J),Q(J)
[illegible]     FORMAT (4X,I[illegible],8X,I2,1[illegible]X,E13.6)
[illegible]   CONTINUE
      PRINT [illegible],[illegible]N(MM)
[illegible]     FORMAT ([illegible]X,*NUMBER OF SET UPS /YEAR = *,E[illegible]3.6)
      PRINT [illegible],TC(MM)
          FORMAT ([illegible]X,*TOTAL COST OF THE SYSTEM = *,[illegible].8)
      IF (KOUNT .LT. [illegible]PROB) GO TO 5[illegible]
      STOP
      END
```

```
C
C*************************************************************************
C
C       THIS PROGRAM WAS USED TO TEST THE PERFORMANCE OF HEURISTIC
C
C*************************************************************************
C
C       A(I) = ORDERING COST FOR ITEM I
C            = FIXED SET UP COST
C       V(I) = VARIABLE SET UP COST FOR ITEM I
C       H(I) = HOLDING COST FOR ITEM I
C       B(I) = BACKORDER COST FOR ITEM I
C       W(I) = WAREHOUSE SPACE OCCUPIED BY ONE UNIT OF ITEM I
C       R    = TOTAL AVAILABLE WAREHOUSE SPACE
C       Q(I) =  ECONOMIC ORDER QUANTITY FOR ITEM I
C
      DIMENSION V(  ),D(  ),H(  ),B(  ),VC(  ,  ),CK(  ),K(  ),F(  ),
     1Q(  ),IK(  ),MK(  ),PK(  ),NPK(  ),X(  ),BK(  ),Q(  ),W(  ),
     2IT(  ),ITSQ(  ),Z(  )
C*************************************************************************
      DO  4 NPROB=1,NPROB
      READ (  )
      KOUNT =
      KOUNT = KOUNT +
      DO  3 M=1,NITEM
      READ (  )
C
      FORMAT (/,  ,*PROBLEM DESCRIPTION*)
      READ (  ,M
      PRINT (/,  ,* NUMBER OF PERIODS /YEAR = *,I4)
      READ (  ,
      FORMAT (2  ,A)
      DO 2    ,S,R,P
      READ (  ,K
      PRINT (  ,*WAREHOUSE COST = *,  .2)
      READ (  ,P
      PRINT (  ,*WAREHOUSE SPACE AVAILABLE = *,F 8.6)
      READ (  ,A,B
      FORMAT (  ,*FIXED SET UP COST = *,  .2)
      PRINT 5
    5 FORMAT (/,  ,*,ITEM NO.          VARIABLE SET UP COST          DEMAND
     1      HOLDING COST          BACKORDER COST       SPACE /UNIT*,/)
      DO  I =  ,NITEM
      READ (  ,D(I),V (I),H(I),B(I),W(I)
      PRINT 9  ,I,VC(  ),D(I),H(I),B(I),W(I)
    9 FORMAT (6X,I2  ,8X,F8.6,6X,F  .6,8X,F  .6,6X,F  .6)
      CONTINUE
      DO  7  I= ,N
      K(I) =
      CK(  ) = K(I)
    7 CONTINUE
C*************************************************************************
      DM = M
      DO =
```

```
*****************************************************************************
      [illegible]
      Y = [illegible]
      DO [illegible] I = [illegible],N
      [illegible] = [illegible]+((DM** [illegible])*D(I)*H(I)*B(I)*CK(I)+R*D(I)*W(I)*CK(I)*([illegible]*DM
     *[illegible])-R*W(I)))/(H(I)+B(I))
      Y = Y + VS(I)/CK(I)
      CONTINUE
      RC = [illegible]/([illegible]*DM*([illegible]+Y))
      [illegible] = SQRT(SQN)
      [illegible] = [illegible]*[illegible]+ [illegible]*Y+ [illegible]/([illegible]* [illegible]) - R*[illegible]
      IF ([illegible]) GO TO [illegible]
      RC = [illegible]
      [illegible] = [illegible] +
C*****************************************************************************
      DO [illegible] I = [illegible],N
      F(I) = [illegible]*DM*VS([illegible])*(H(I)+B(I))/(D(I)*((DM**[illegible])*H(I)*B(I)
     +R*W(I)*([illegible]*DM*[illegible])-R*W(I)))
      G(I) = SQRT(F(I))
      [illegible]/([illegible]) = [illegible]*G([illegible])
      [illegible]
      DO [illegible] I = [illegible],N
      IF (BK([illegible]) - [illegible]) [illegible]
      [illegible]K([illegible]) =
      [illegible]
      [illegible]K([illegible]) = [illegible]K(I)
      [illegible]K(I) = NK(I)
      VS([illegible]) = ([illegible]*V[illegible](I))/ [illegible]K(I)+(D(I)*DM*H(I)*B(I)*AK(I))/([illegible]*[illegible]N*
     (H([illegible])+B([illegible]))) + R*W(I)*D(I)*[illegible]K([illegible])*([illegible]*DM*B([illegible])-R*W(I))/
     ([illegible]*DM*[illegible]*(H(I)+B(I)))
      [illegible]K([illegible]) = [illegible]K([illegible])+
      BK([illegible]) = MK(I)
      VS([illegible]) = ([illegible]N*V[illegible](I))/BK(I)+(D(I)*DM*H(I)*B(I)*BK(I))/([illegible]*[illegible]N*
     (H([illegible])+B([illegible]))) + R*W(I)*D(I)*BK(I)*([illegible]*DM*B(I)-R*W(I))/
     ([illegible]*DM*[illegible]*(H(I)+B([illegible])))
      IF (VS([illegible])-VS([illegible])) [illegible]
    [illegible] NBK(I) = NK([illegible])
      GO TO [illegible]
      NBK(I) = MK(I)
    [illegible] CONTINUE
      DO [illegible] I = [illegible],N
      IF (NBK(I) [illegible] K(I)) GO TO [illegible]
      CONTINUE
      GO TO [illegible]
    [illegible] DO [illegible] I = [illegible],N
      K(I) = NBK(I)
      CK([illegible]) = K(I)
    [illegible] CONTINUE
      GO TO [illegible]
      DO [illegible] I = [illegible],N
      Q([illegible]) = D(I)*CK([illegible])/[illegible]N
    [illegible] CONTINUE
      DO V[illegible] = (RC - TC)*[illegible]/RC
C*****************************************************************************
```

```
      [illegible]
    [illegible] FORMAT (/,[illegible],*OPTIMUM VALUES ARE AS FOLLOWS*)
      [illegible]
      FORMAT (/,[illegible] *ITEM NO.        VALUE OF K       ORDER QUANTITY *,/)
      [illegible] I= [illegible],[illegible]
      [illegible],K(I),Q(I)
    [illegible] FORMAT ([illegible],I4,[illegible],I4,[illegible],[illegible].6)
      [illegible]
      [illegible] N
    [illegible] FORMAT ([illegible],* OPTIMUM NUMBER OF SET UPS = *,[illegible])
      [illegible],IT
    9 FORMAT ([illegible],*TOTAL NO. OF ITERATIONS = *,I4)
      [illegible],TC
    [illegible] FORMAT ([illegible],* TOTAL COST OF THE SYSTEM = *,[illegible]6-8)
      PRINT [illegible],PSAV
      FORMAT ([illegible],*PERCENTAGE SAVING = *,[illegible]4.8)
C*****************************************************************************
      IF (KOUNT.GET.[illegible]PROB) GO TO [illegible]
      STOP
      [illegible]
```

```
C
C**********************************************************************
C     MAIN LINE PROGRAM FOR DISCRETE DYNAMIC PROGRAMMING
C**********************************************************************
C
C**********************************************************************
C     D(I) = DEMAND RATE FOR ITEM I
C     R(I) = REPLENISHMENT RATE FOR ITEM I
C     HC(I) = HOLDING COST FOR ITEM I
C     BC(I) = BACKORDER COST FOR ITEM I
C     TL(I) = LEAD TIME FOR ITEM I
C     C(I) = ITEM COST FOR ITEM I
C     PC(I) = PROCUREMENT COST FOR ITEM I
C     W(I) = WAREHOUSE SPACE OCCUPIED BY ON UNIT OF ITEM I
C     WR = TOTAL AVAILABLE WAREHOUSE SPACE
C     IPRCB = 0 FOR FINITE REPLENISHMENT RATE
C     IPRCB = 1 FOR INFINITE REPLENISHMENT RATE
C**********************************************************************
C
      DIMENSION D(  ),HC(  ),BC(  ),TL(  ),C(  ),PC(  ),IW(  ),X(4,  ),
     1G(  ,  ),PL(  ,  ),F(4,  ),G(4,  ),FMIN(4,  ),FMIN(4,  ,2),
     2JJ(  ,  ),IOPT(  ),IDEC(  ),LMN(4,  ),FMINI(4,  ),RR(  )
C**********************************************************************
      READ(  ,  ) NITEM,IP
      IPRCB =
      READ(  ,  ) (D(I),I=1,NITEM)
      READ(  ,  ) (R(I),I=1,NITEM)
      READ(  ,  ) (HC(I),I=1,NITEM)
      READ(  ,  ) (BC(I),I=1,NITEM)
      READ(  ,  ) (TL(I),I=1,NITEM)
      READ(  ,  ) (C(I),I=1,NITEM)
      READ(  ,  ) (PC(I),I=1,NITEM)
      READ(  ,  ) (W(I),I=1,NITEM)
      READ(  ,  ) WR
      WRITE(  ,  )
      WRITE(  ,  )
      WRITE(  ,  )
```

```
      PQ(I,J) = SQRT((2.*PC(I)*D(I)*(1-D(I)/RR(I))+(X(I,J)**2)*
     (HC(I)+BC(I)))/BC(I))*2./(1-D(I)/RR(I))
      PL(I,J) = X(I,J)+D(I)*TL(I)-PQ(I,J)*(1-D(I)/RR(I))
      F(I,J)=C(I)*D(I)+PC(I)*D(I)/PQ(I,J)+HC(I)*(X(I,J)**2)/(2.*PQ(I,J)*
     (1-D(I)/RR(I)))+(BC(I)*(D(I)*TL(I)-PL(I,J))**2)/(2.*PQ(I,J)*
     (1-D(I)/RR(I)))
      GO TO
      PQ(I,J) = SQRT((2.*PC(I)*D(I)+(X(I,J)**2)*(HC(I)+BC(I)))/BC(I))
      PL(I,J) = X(I,J)+D(I)*TL(I)-PQ(I,J)
      F(I,J)=C(I)*D(I)+PC(I)*D(I)/PQ(I,J)+HC(I)*(X(I,J)**2)/(2.*PQ(I,J))
     +(BC(I)*(D(I)*TL(I)-PL(I,J))**2)/(2.*PQ(I,J))
      G(I,J) = F(I,J)
      CONTINUE
C*****************************************************************************
      PRINT 24
      FORMAT (//////,1X,*ANSWER TO THE ABOVE PROBLEM IS GIVEN BELOW*)
      PRINT 1
    2 FORMAT (//,1X,*TABLE OF VALUES FOR STAGE  *)
      PRINT 9
    9 FORMAT (////  1X,*STAT  VALU       FUNCTION VALUE      DECISION VALUE
     *,/)
C*****************************************************************************
      IW  = IW( )
      LP =
      DO    J =  ,IP, W
      JH = J-
      FMIN( ,J) = F( ,J)
      JJ( ,J) = J-
      LP = LP+
      LM ( ,LP) = J
      FMIN( ,LP) = FMIN( ,J)
      PRINT   ,JH,FMIN( ,J),JJ( ,J)
      FORMAT (6X,I ,9X,F 1.6,  X,I )
      CONTINUE
C*****************************************************************************
      DO    I= ,N
      PRINT   ,I
    1 FORMAT (////,1X,*TABLE OF VALUES FOR STAGE*,I )
      PRINT
      FORMAT (////,1X,*STAT. VALU.      FUNCTION VALUE      DECISION VALUE
     *,/)
      IWJ = IW(I)
      LK =LP
      LP =
      DO    J= ,IR
      JS = J-
      II = I-
      LM = J-IW(II)
      IF (LM .GE. 1) GO TO 1
      IWI = IWJ
      JJ = JS
      II = JJ/IWI
      IA = IA
      RR = IA
```

```
      IF (RB - [illegible]) [illegible]
      FMIN(I,J) = G(I,J) + FMIN(II, [illegible])
      JJ(I,J) = JS
      LP = LP +
      FMIN1(I,LP) = FMIN(I,J)
      LMN(I,LP) = J
      [illegible]
      MK =
      DO [illegible] IM = [illegible],LK
      IYJ = J-LMN(II,IM)
      IF(IYJ) [illegible]
      YWI = IWJ
      LMNI = LMN(II,IM)
      YJ = IYJ
      BB = YJ/YWI
      IB = BB
      RB = IB
      IF (RB - BB) [illegible]
      IIYJ = IYJ +
      MK = MK+
      FM(I,J,MK) = G(I,IIYJ) + FMIN(II,LMNI)
      IDC(MK) = IYJ
      CONTINUE
      IF (MK .LT. [illegible]) GO TO 4
      LP = LP+
      LMN(I,LP) = J
      MS =
      MT = MS
      IF (MT .Q. MK) GO TO 4
      MS = MT+
      IF (FM(I,J,MT) .LT. FM(I,J,MS)) GO TO 7
      GO [illegible]
      IF (MS .Q. MK) GO TO 4
      MS = MS+
      GO TO [illegible]
      FMIN(I,J) = FM(I,J,MT)
      FMIN1(I,LP) = FMIN(I,J)
      JJ(I,J) = IDC(MT)
C*****************************************************************************
      PRINT [illegible],JS,FMIN(I,J),JJ(I,J)
      FORMAT (L2,I[illegible],9X,[illegible],I[illegible])
C*****************************************************************************
      CONTINUE
      CONTINUE
C*****************************************************************************
      IWN = IW(N)
      MX =
      MY = MX
      IF (MY .Q. LP) GO TO [illegible]
      MX =MX+
      IF (FMIN1(N,MY) .LT. FMIN1(N,MX)) GO TO [illegible]
      GO TO [illegible]
      IF (MX .Q. LP) GO TO [illegible]
      MX =MX+
```

```
      GO TO 1
      FOPT = FMIN1(N,MY)
C*****************************************************************************
      PRINT [illegible],FOPT
      FORMAT (////,[illegible],*TOTAL COST OF THE SYSTEM = *,E 3.6)
C*****************************************************************************
      IP = N
      JP = LMN(IP,MY)
      ICPT(IP) = JJ(IP,JP)
C*****************************************************************************
      PRINT [illegible],IP,ICPT(IP)
      FORMAT (/,[illegible],*TH  WAREHOUSE SPAC  ALLOTED TO ITEM NO.*,I2,* = *
     ,I2)
C*****************************************************************************
      JP = JP -JJ(IP,JP)
      IP = IP -[illegible]
      IF (IP .GT. [illegible]) GO TO  9
C*****************************************************************************
      STOP
      END
```